Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Matevž Lenič

Podporna aplikacija za strelska društva

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: doc. dr. Aleš Smrdel

Ljubljana 2015

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge: Podporna aplikacija za strelska društva

V diplomskem delu razvijte spletno aplikacijo, ki bo služila kot podpora strelskim društvom. Aplikacija naj omogoča vlogo navadnega uporabnika, ki ima omejen nabor možnosti, in pa vlogo strelskega društva, ki ima poln nabor možnosti. Tako naj ima uporabnik možnost prijave, pregleda novic, pravil in strelišč, prav tako pa naj ima na voljo tudi nadzorno ploščo, ki naj mu omogoča beleženje rezultatov, prijavo na turnirje in pregled statistik. Strelsko društvo pa naj ima še možnost razpisa in izvajanja različnih turnirjev, pregled rezultatov izvedenega turnirja ter sprejem novih članov. Poleg tega pa razvijte tudi aplikacijo za mobilno platformo Android, ki bo uporabniku omogočala slikanje tarč in pošiljanje slik tarč na strežnik, ki bo te slike shranil v bazo za prikaz in nadaljnjo obravnavo. Pri realizaciji aplikacije izberite tiste tehnologije na strani strežnika in na strani odjemalca, ki bodo omogočale hitro in odzivno delovanje aplikacije. Prav tako pa poskrbite za dobro uporabniško izkušnjo pri uporabi aplikacije na napravah z različnimi dimenzijami zaslonov.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Matevž Lenič, z vpisno številko **63110290**, sem avtor diplomskega dela z naslovom:

Podporna aplikacija za strelska društva

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Aleša Smrdela,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 14. septembra 2015 Podpis avtorja:

Zahvaljujem se mentorju doc. dr. Alešu Smrdelu za strokovno pomoč in ponujene nasvete pri izdelavi diplomskega dela.
Zahvaljujem se vsem bližnjim, ki so me tekom študija podpirali.
Zahvaljujem se tudi kolegoma Maticu Tribušonu in Žigi Lesarju, ki sta mi v času študija ponudila mnogo strokovnih nasvetov.

Kazalo

Seznam uporabljenih kratic

kratica	angleški pomen	slovenska razlaga
HTML	HyperText Markup Language	Označevalni jezik za oblikovanje večpredstavnostnih dokumentov
PHP	Hypertext Preprocessor	Skriptni jezik za izdelavo spletnih strani
CSS	Cascading Style Sheets	Kaskadne slogovne pole
W3C	World Wide Web Consortium	Organizacija za spletne standarde
SQL	Structured Query Language	Strukturirani povpraševalni jezik za delo s podatkovnimi bazami
DOM	Document Object Model	Objektni model dokumenta
SVG	Scalable Vector Graphics	Umerljiva vektorska grafika
VML	Vector Markup Language	Vektorski označevalni jezik
ADT	Android Development Tools	Orodja za razvijanje Android aplikacij
ORM	Object-Relational Mapping	Mapiranje objektnih razmerij
MVC	Model-View-Controller	Model-pogled-krmilnik
REST	Representational State Transfer	Prenos predstavitvenega stanja
HTTP	The Hypertext Transfer Protocol	Protokol za izmenjavo hiperteksta
URI	Uniform Resource Identifier	Unikaten identifikator vira
API	Application Programming Interface	Programski vmesnik aplikacije
XML	EXtensible Markup Language	Razširljiv označevalni jezik
AJAX	Asynchronous JavaScript and XML	Asinhroni JavaScript in XML
JSON	JavaScript Object Notation	JavaScript objekt za prenašanje podatkov

CSRF	Cross-Site Request Forgery	Ponarejanje spletnih zahtev
RGB	Red, Green, Blue color model	Barvni model RGB (rdeča, zelena, modra)

Povzetek

Beleženje strelskih rezultatov je, sploh brez elektronske podpore, lahko zelo mučno delo, za katerega še ni neke praktične rešitve. Zato smo se odločili izdelati podporno aplikacijo za strelska društva. Aplikacija, poleg beleženja rezultatov, ponuja še mnoge druge uporabne funkcionalnosti za strelce in strelska društva. Poleg spletne aplikacije pa smo razvili tudi aplikacijo za mobilno platformo Android, ki strelcu omogoča slikanje tarč in prenašanje slik na strežnik za kasnejši pregled. Prvi del diplomske naloge vsebuje opis tehnologij, ki smo jih uporabili za razvoj aplikacij. Drugi del pa vsebuje opis problema, temu sledi opis implementacije.
Za razvoj spletne aplikacije smo na strežniški strani sistema uporabili jezik PHP v kombinaciji z ogrodjem Laravel. Za zagotovitev večje interaktivnosti spletne strani pa smo na strani odjemalca uporabili JavaScript. Za hranjenje podatkov smo uporabili podatkovno bazo MySQL. Mobilna aplikacija je bila napisana v programskem jeziku Java.

Ključne besede: spletna aplikacija, mobilna aplikacija, strežnik-odjemalec, REST, Android, Laravel, PHP.

Abstract

The process of logging shooting results, especially without proper electronic support, can be a very tedious task, for which no practical solution exists. The web application, in addition to logging results, offers many other useful functionalities to shooters and to shooting clubs. In addition to the web application, we also developed an Android mobile application, which enables shooters to take photos of their targets and upload them to a server so they can review them later. The first part of the thesis contains the description of the technology, frameworks and tools which we used for the development of the application. The second part contains the description of the problem, followed by the description of the implementation of web and mobile application. For the development of the web application on the server side we used the PHP programming language in combination with Laravel framework. To make the web application more interactive, we used JavaScript on the client side. MySQL database was used to store data. Mobile application was written in programming language Java.

Keywords: web application, mobile application, client-server, REST, Android, Laravel, PHP.

Poglavje 1

Uvod

Strelstvo je olimpijska športna disciplina, ki je v Sloveniji precej razširjena. Samo v Sloveniji imamo preko 50 strelskih društev, ki vsakodnevno prirejajo treninge in tekmovanja. S pojmom strelstvo imamo lahko v mislih več stvari, mi pa se bomo v naši diplomski nalogi osredotočili na panoge, pri katerih tekmovalci uporabljajo strelno orožje za streljanje v tarče. Problem nastane, ko si strelci želijo beležiti vse rezultate treningov in turnirjev, saj večina treningov obsega streljanje v papirnate tarče. Večina elektronskih sistemov pa je preveč zastarelih, da bi pripomogli k izboljšanju kakovosti treninga. Namen naše diplomske naloge je izdelava platforme, ki bi združila skupaj vsa strelska društva in in jim olajšala organizacijo turnirjev, obenem pa bi strelcem omogočila enostavnejši pregled in prijavo na turnirje ter olajšala treninge. S tem namenom smo se odločili izdelati spletno aplikacijo v jeziku PHP z ogrodjem Laravel ter mobilno aplikacijo v jeziku Java za operacijski sistem Android. Mobilna aplikacija strelcu omogoča slikanje tarč in prenašanje slik na strežnik za kasnejši pregled.

1. Cilji spletne aplikacije
   Glavni cilj spletne aplikacije je narediti platformo, ki bo strelcem pomagala pri njihovi strelski poti. Ta cilj želimo doseči tako, da:

   - Ponudimo strelcem enotno aplikacijo, kjer so dostopne vse infor-

macije, ki jih potrebujejo.

- Omogočimo strelcem lažje beleženje rezultatov treninga.
- Omogočimo strelcem mobilno podporo pri beleženju rezultatov.
- Omogočimo strelcem pregled in analizo njihovih rezultatov.
- Omogočimo strelcem enoten pregled vseh turnirjev.
- Omogočimo strelskim društvom prirejanje turnirjev in vodenje evidence članov.

To so glavni cilji, ki jim mora aplikacija zadostiti, še zdaleč pa niso vsi.

2. Pregled obstoječih aplikacij
   Aplikacije, kot je naša, na tržišču še ni. Obstajajo različne strelske strani, vendar nobena stran ne omogoča beleženje rezultatov in prikazovanje statistik. Večina strani, ki so namenjene strelcem, je le informativne narave in ne omogoča željene interaktivnosti, kar je konkurenčna prednost razvite aplikacije.

3. Prednosti aplikacije
   Kot je bilo že omenjeno v prejšnjem odstavku, je naša aplikacija, za razliko od ostalih, interaktivna. Ker je nadzorna plošča, v kateri lahko uporabnik beleži rezultate, prilagojena za mobilne telefone, je naša aplikacija še toliko bolj uporabna. Aplikacija ima tudi mobilno podporo, kar je dandanes zelo pomembno.
   Aplikacija je narejena za vse starostne skupine, zato ima tudi nezahteven uporabniški vmesnik.

Poglavje 2

Tehnologija

To poglavje predstavlja osnovne tehnologije, s katerimi smo ustvarili našo spletno aplikacijo. Opisane tehnologije smo glede na njihove značilnosti razporedili v tri skupine. Prva skupina vsebuje tehnologije kot so HTML, PHP, JavaScript ter CSS. Gre za osnovne programske in označevalne jezike, ki jih razvijalci zelo pogosto uporabljajo in so za razvoj spletnih aplikacij izjemnega pomena. Druga skupina tehnologij so programske knjižnice in ogrodja [1]. Programske knjižnice so zbirke funkcij, ki jih lahko razvijalci uporabljajo neodvisno od aplikacij, ki jih delajo, in nam zelo olajšajo programiranje. Funkcije knjižnic se obnašajo enako v kateremkoli programu, ki jih pokliče. Primer ene izmed najbolj razširjenih programskih knjižnic pri izdelavi spletnih strani je jQuery. Programsko ogrodje je neka množica programske kode ali knjižnic, ki nam ponuja lažji razvoj spletnih aplikacij. Tipično nam ena programska knjižnica ponavadi ponuja neko specifično funkcionalnost, medtem ko nam ogrodje ponuja širšo paleto funkcionalnosti, s katerimi si lahko pomagamo. Običajno je ogrodje sestavljeno iz več plasti [2], ki nam povedo, kako naj bi bila aplikacija sestavljena. Ker se programerjem pri izdelavi aplikacije ni potrebno ukvarjati z vzpostavitvijo delujočega sistema na nižjem nivoju, pač pa se lahko takoj lotijo zadovoljevanja naročnikovih zahtev, se zmanjša čas izdelave aplikacije. Pri izdelavi naše aplikacije smo uporabili ogrodje za izdelavo PHP aplikacij Laravel [3].

Tretja skupina tehnologij so orodja. Gre za aplikacije, ki ponavadi vsebujejo nek grafični uporabniški vmesnik. Primer aplikacije, ki spada v to skupino, je urejevalnik besedil Sublime, ki vsebuje več sistemov za pomoč programerju, kot sta dopolnjevanje kode in označevanje sintakse.

2.1 Tehnologije

2.1.1 HTML

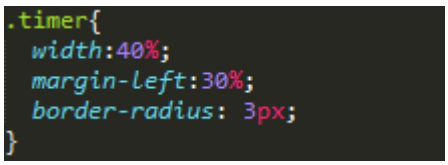
HTML je označevalni jezik za oblikovanje večpredstavnostnih dokumentov [4]. HTML je od leta 1990, ko je bil predstavljen, do leta 2015 postal standarden jezik za izdelovanje spletnih strani. Vsaka stran lahko vsebuje povezave do katerekoli strani, ki se nahaja na internetu. Celoten HTML dokument je sestavljen iz teksta in HTML kode. Značke so tisto, kar razlikuje HTML kodo in navaden tekst. Na sliki 2.1 je prikazana tipična uporaba HTML značk v dokumentu. Značke nam omogočajo, da na strani uporabljamo veliko različnih stvari, od različnih slogov teksta, do tabel. Za standardizacijo HTML jezika skrbi W3C [5]. HTML dokumente lahko pregledujemo v spletnih brskalnikih, ki poskrbijo za izris teksta in drugih elementov na strani v obliki, kot smo si jo zaželeli. Ker je jezik standardiziran, je večina njegovih sintaktičnih elementov prepoznanih v vseh brskalnikih. Konec leta 2014 je bila izdana najnovejša različica jezika HTML5, ki podpira še večjo uporabo multimedijskih vsebin.

```
<div class="title">
    <h1>Združena strelska društva</h1>
    <p>Dobrodošli v naši spletni aplikaciji</p>
</div>
```

Slika 2.1: Prikaz HTML značk v dokumentu.

2.1.2 CSS

Kaskadne slogovne pole določajo način kontrole predstavitve HTML dokumentov [6]. So preprost način dodajanja slogov spletnim dokumentom. Običajno je slogovna pola ločena od HTML dokumenta. Ta ločitev nam prinese veliko dobrih stvari, kot sta lažja dostopnost vsebine ter manj ponavljanja kode v HTML dokumentu. Prav tako lahko več različnih dokumentov uporablja isto kaskadno slogovno polo. Ni pa nujno, da so definicije sloga vedno izven HTML dokumenta. Možno je tudi, da je slog definiran v istem dokumentu, in sicer kot vrstična definicija, ki določa slog za posamezen element, lahko pa je slog definiran na nivoju dokumenta, tako, da določa stil za vse elemente znotraj dokumenta. Imamo več načinov določanja sloga, najpogostejša sta določanje glede na razred oziroma identifikator elementa. Slika 2.2 prikazuje primer določitve sloga glede na razred elementov. Najnovejši standard, ki je združljiv s prejšnjimi standardi, je CSS3, ki ga standardizira in razvija W3C.



Slika 2.2: Prikaz CSS slogovnega pravila z definiranjem generičnega razreda sloga.

2.1.3 PHP

PHP je zelo popularen odprtokoden skriptni jezik, ki je bil zasnovan za razvijanje spletnih aplikacij. Jezik se je začel razvijati leta 1994, sedaj pa za njegov razvoj skrbi *The PHP Group* [7]. PHP stran vsebuje HTML z vgrajeno PHP kodo, ki je interpretirana preden se stran pošlje odjemalcu. Odjemalec dobi le rezultat interpretirane kode, ker pa se je koda izvedla na strani strežnika, ne more vedeti kakšna je ta koda bila. PHP koda je vstavljena med posebni

začetni in končni znački, ki definirata, kdaj se mora izvajati PHP koda in kdaj ne. Slika 2.3 prikazuje primer PHP kode, ki se interpretira na strani strežnika.
Kljub svoji starosti, je jezik PHP še danes ena izmed strežniških tehnologij na več kot 80 odstotkih straneh [9].

```php
<?php
    echo "Dobrodošli na spletni aplikaciji!";
?>
```

Slika 2.3: Primer PHP kode, ki se interpretira pred pošiljanjem.

2.1.4 JavaScript

JavaScript je programski jezik [10], ki se ga v največji meri uporablja za interaktivnost spletnih strani. Prav tako kot PHP je tudi JavaScript jezik, ki se interpretira. Glavna razlika med njima je, da se PHP interpretira na strani strežnika, JavaScript pa na strani odjemalca. To je dobra lastnost, saj se razbremeni strežnik in bolj obremeni odjemalca. JavaScript kodo lahko vključimo v HTML dokument s pomočjo značke $< script > ... < /script >$, lahko pa kot pri CSS naredimo zunanjo datoteko, ki jo nato vključimo v dokument. Primer JavaScript zanke je prikazan na sliki 2.4. V zadnjem času so v porastu tudi različna JavaScript ogrodja, kot so Angular, Ember in Backbone, ki naredijo JavaScript še veliko bolj uporaben. JavaScript pa ni omejen samo na stran uporabnika, možno ga je uporabiti tudi na strani strežnika, primer je Node.js. JavaScript podpira večina novejših brskalnikov.

```javascript
for (var i = 0; i < data.length; i++) {
    console.log(i);
}
```

Slika 2.4: Primer for zanke v jeziku JavaScript.

2.1.5 Java

Java je programski jezik, ki so ga leta 1995 razvili pri Sun Microsystems [11]. Izdelan je bil po principu *Napiši enkrat, poženi povsod* [12]. To pomeni, da je programski jezik neodvisen od platforme, kjer kodo poganjamo. To pomeni, da lahko program poženemo kjerkoli, brez da bi ga morali ponovno prevesti. Jezik Java lahko uporabimo za razvoj različnih aplikacij, od aplikacij, ki jih poganjamo na enem računalniku, do aplikacij, ki so porazdeljene po različnih strežnikih in omrežjih. Java je tretja najbolj razširjena tehnologija, ki se uporablja pri izdelovanju spletnih strani. Je precej manj razširjena kot PHP ali .NET vendar je zelo robustna in se jo zato v večini uporablja na straneh preko katerih poteka veliko prometa [13]. V jeziku Java se programirajo tudi aplikacije za operacijski sistem Android. Slika 2.5 prikazuje primer funkcije napisan v jeziku Java.

```
public static void printTabele(int[] elementi){
    for(int i = 0; i < elementi.length; i++){
        System.out.print(elementi[i]);
    }
    System.out.println();
}
```

Slika 2.5: Primer funkcije v jeziku Java.

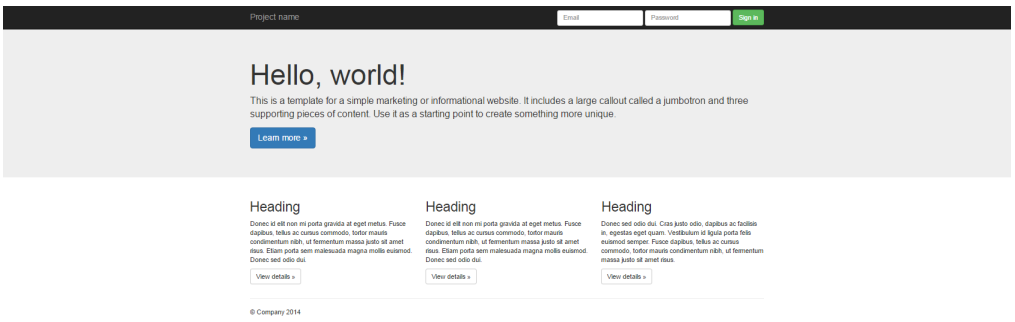
2.1.6 MySQL

Za uspešno delovanje spletne aplikacije, moramo ponavadi tudi shranjevati podatke. Med temi podatki lahko opišemo tudi relacije. Podatke nato shranimo v podatkovne zbirke. Da te podatke lahko ustvarjamo, vzdržujemo in nadzorujemo, potrebujemo nek sistem za upravljanje s podatkovnimi bazami. Eden izmed sistemov za upravljanje s podatkovnimi bazami je MySQL [14], ki je odprtokodna implementacija relacijske podatkovne baze in je trenutno najbolj uporabljana. Za delo s podatki uporablja SQL, ki je najbolj razširjen in standardiziran povpraševalni jezik za delo s podatki.

2.2 Knjižnice in ogrodja

2.2.1 Bootstrap

Bootstrap je odprtokodno JavaScript ogrodje, ki so ga leta 2011 razvili pri podjetju Twitter. Na začetku se je imenoval Twitter Bootstrap, nato so ga preimenovali v Bootstrap. Uporablja se za izdelavo spletnih strani in aplikacij. Trenutno se uporablja različica 3, v kratkem pa prihaja že različica 4, ki naj bi še bolj olajšala izdelavo spletnih strani. Knjižnica vsebuje vnaprej definirane HTML in CSS predloge za različne komponente spletnih strani, kot so gumbi in navigacijske vrstice [15]. Cilj ogrodja je olajšanje razvijanja aplikacij na strani odjemalca in večja interaktivnost.
Ogrodje nam prav tako ponuja možnosti za izdelavo odzivnih aplikacij. To pomeni, da se aplikacija prilagaja vsem vrstam naprav, od mobilnih do stacionarnih. Dodatna prednost je tudi preprostost uporabe. Vse kar mora razvijalec narediti, je obiskati njihovo stran, prenesti njihovo kodo, jo namestiti v svoj projekt in že je pripravljeno na uporabo. Ker je ogrodje v zadnjih letih postalo zelo razširjeno, so se na spletu pojavile mnoge strani, kjer so objavljene različno oblikovane teme in komponente, ki jih lahko razvijalec uporabi. Slika 2.6 prikazuje primer teme narejene z ogrodjem Bootstrap 3, ki smo ga uporabili pri razvoju naše aplikacije.



Slika 2.6: Primer prostodostopne teme, narejene z ogrodjem Boostrap 3.

2.2.2 jQuery

jQuery je JavaScript knjižnica, ki je bila izdelana z namenom, da olajša izdelavo kode, ki se izvaja na strani odjemalca [16]. Je prostodostopna, odprtokodna knjižnica in najbolj priljubljena JavaScript knjižnica, ki jo uporablja več kot 65 odstotkov strani z največ prometa na svetu. V sami osnovi je jQuery knjižnica, s katero manipuliramo DOM elemente. DOM je predstavitev dokumenta oziroma spletne strani, kjer so vsi elementi spletnega dokumenta, zapisani hierarhično v drevesni strukturi. jQuery nam poenostavi sintakso za iskanje, izbiranje in uporabljanje teh elementov. To naredi tako, da neke osnovne naloge, za katere bi morali v jeziku JavaScript napisati veliko vrstic, zavije v metode, ki jih lahko kličemo z eno samo vrstico. jQuery uporablja veliko največjih firm, kot so Microsoft, Google in IBM. Podprt je v vseh novejših spletnih brskalnikih. Slika 2.7 prikazuje primer manipulacije HTML elementov z uporabo knjižnice jQuery.

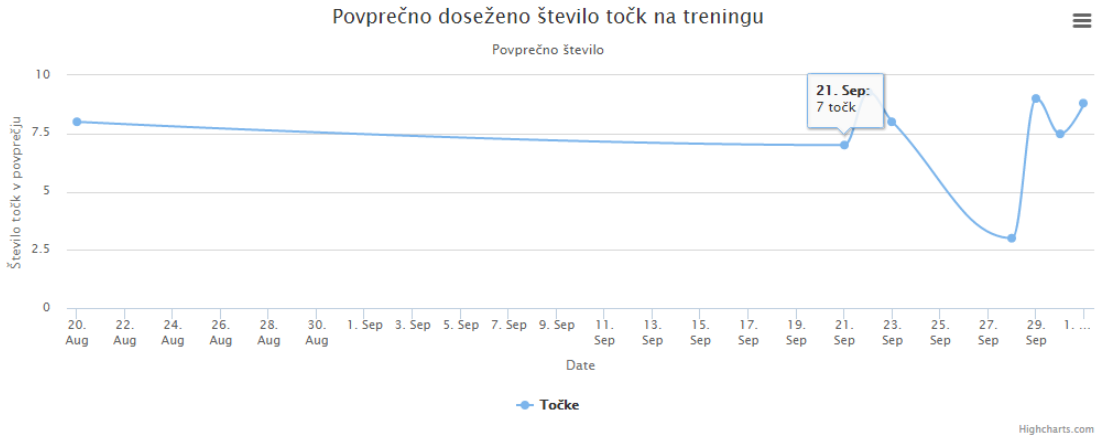
```
$('#dp1').datepicker({
    format: "yyyy-mm-dd"
});

$("#dashBoardDivTraining").empty();
$("#dashBoardDivTraining").hide();

var parent = $("#trainingDiv").parent();
if(!parent.hasClass("active")){
    parent.toggleClass('active');
}
```

Slika 2.7: Primer manipulacije HTML elementov s knjižnico jQuery.

2.2.3 Highcharts

Highcharts je JavaScript knjižnica, ki ponuja izris grafov na spletnih straneh. Je zelo močno orodje, saj nam ponuja izris prek dvajset različnih vrst grafov, od črtnega grafa do stolpičastega. S pomočjo tehnologij HTML, SVG in VML, ki jih uporablja za izris, nam knjižnica izriše zelo lične in interaktivne grafe. Slika 2.8 prikazuje primer grafa, izrisanega z uporabo knjižnice Highcharts. Ena izmed prednosti je, da je knjižnica odprtokodna in jo lahko

razvijalec po želji in potrebi spreminja. Knjižnico podpirajo vsi novejši brskalniki. Omogoča nam tudi izvoz grafov v formatih PNG, JPG, PDF in SVG. Uporaba knjižnice je za osebne, izobraževalne in neprofitne namene prosta, za profitno uporabo pa je treba kupiti licenco. Izmed podjetij, ki se nahajajo na prestižni Fortune 100 lestvici najbogatejših podjetij na svetu, jih kar 61 uporablja knjižnico Highcharts [17].



Slika 2.8: Primer črtnega diagrama izrisanega s knjižnico Highcharts.

2.2.4 Laravel

Laravel je odprtokodno ogrodje za izdelovanje spletnih aplikacij v jeziku PHP. Leta 2011 ga je ustvaril Taylor Otwell [18]. Kljub temu, da je na tržišču šele štiri leta, pa je od takrat izšlo že pet verzij, trenutna najnovejša je verzija 5.1. Od leta 2011 je Laravel ogrodje postalo najbolj uporabljano ogrodje za jezik PHP [19] in je prehitelo celo dve precej starejši ogrodji Codeigniter in Symphony. Kar je ogrodju pomagalo pri uspehu, je lahka namestitev na vseh popularnih platformah. Vse, kar razvijalec potrebuje, je upravljavca paketov za PHP Composer, kjer je potrebno zagnati le nekaj vrstic v vmesniku z ukazno vrstico, in že je ogrodje pripravljeno za uporabo. Da pa so razvijalci Laravela naredili ogrodje še bolj privlačno, so napisali skoraj perfektno in v praksi zelo uporabno dokumentacijo [3], kjer lahko razvijalec spletne aplika-

cije najde praktično vse primere uporabe funkcij ogrodja. Če razvijalcu še to ni dovolj, pa ima na voljo še Laracasts. Tam lahko razvijalec najde več kot 500 različnih posnetkov, kjer so prikazane vse funkcionalnosti, ki jih Laravel ponuja.
Laravel ponuja več funkcionalnosti, zaradi katerih je postal najbolj uporabljano PHP ogrodje:

1. **ORM** *(angl. Object-relational mapping)*
   ORM je orodje, ki nam pri razvoju olajša delo. Je programska tehnika, ki nam poveže objektno programsko kodo in relacijsko bazo, ki jo imamo v ozadju [20]. Objektna koda je napisana v objektno usmerjenem jeziku, kot je v primeru naše aplikacije jezik PHP. Omogoča nam tudi dostop in spreminjanje mapiranih objektov, pri tem pa nam ni potrebno razmišljati, kako se ti objekti v podatkovni bazi povezujejo. ORM nam zakrije spremembe v podatkovni bazi in če se nam vir podatkov slučajno spremeni, je potrebno spremeniti le ORM in ne aplikacije, ki ga uporablja. Velika prednost je tudi ta, da nam v kodo ni potrebno vključevati SQL stavkov, saj ORM sam poskrbi za njihovo tvorbo. Razvijalec naslavlja le objekte, ki predstavljajo neko entiteto v bazi. Na sliki 2.9 je prikazan primer uporabe orodja ORM za shranjevanje podatkov v bazo.

```
public function saveTraining(){
        $user = User::where('username','=',Session::get('username'))->first();

        $training = new Training;
        $training->name = Input::get('name');
        $training->user_id = $user->id;
        $training->date = date("Y-m-d");
        $training->duration = Input::get('duration');
        $training->averageScore = Input::get('score')/Input::get('numOfShots');

        $saved = $training->save();

}
```

Slika 2.9: Vzorčni primer funkcije, ki nam v ogrodju Laravel s pomočjo ORM shrani trening.

Velika prednost ORM je tudi, da nas prisili, da aplikacijo napišemo v MVC arhitekturi.

2. **MVC** *(angl. Model-View-Controller)*
   MVC je ena izmed arhitektur za razvoj uporabniških vmesnikov in spletnih aplikacij. Zagotavlja nam ločitev med poslovno logiko in predstavitveno logiko. V primeru ogrodja Laravel se kot poslovno logiko običajno definira podatkovne modele, predstavitveno logiko pa predstavlja spletna stran [21].

   - Model
     Predstavlja domeno, okrog katere smo zgradili aplikacijo. Običajno gre za nek realen problem, v našem primeru sta dva izmed modelov predstavljala uporabnika oziroma orožje, ki sta bila oba v relacijski podatkovni bazi definirana kot tabeli.
   - Pogled
     Zadolžen je za izris predstavitvene strani glede na spremembe v modelu.
   - Krmilnik
     Skrbi za akcije, ki se zgodijo ob uporabi spletne aplikacije. Ponuja nam povezavo med modelom in pogledom. Prek krmilnika lahko dostopamo do podatkov, ki jih predstavlja model in jih tudi spreminjamo. Po izvedeni akciji znotraj krmilnika strežniku povemo, kaj je sedaj potrebno narediti. Krmilnik vsebuje nekaj osnovnih akcij, kot sta na primer izdelovanje pogleda oziroma preusmeritev na druge strani. Ogrodje Laravel pa podpira tudi nekatere naprednejše akcije, kot sta na primer posredovanje datoteke ali preusmeritev na druge krmilnike.

   Vse tri komponente so med seboj povezane. Model shranjuje podatke, ki jih lahko pridobimo s krmilnikom in prikažemo v pogledu. Pogled nato izriše predstavitveno stran glede na spremembe v modelu.

3. **RESTful krmilniki**

   REST je arhitektura, ki se uporablja za izdelavo spletnih storitev. Po nekaj več kot desetletju njegove uporabe, je REST postal ena izmed najpomembnejših tehnologij na področju razvoja spletnih aplikacij. Njegova uporaba je najbolj razširjena med skalabilnimi in preprostimi aplikacijami, vendar pa ni omejen samo na to, saj večina REST storitev za svoje delovanje uporablja HTTP protokol, ki je danes prisoten povsod. Strežnik in odjemalec prek HTTP protokola komunicirata z metodami, ki jih protokol uporablja.

   Najbolj uporabljani metodi, ki ju lahko uporabljamo tudi v ogrodju Laravel, sta GET in POST. Preko GET metode naj bi uporabnik le pridobival podatke, prek metode POST pa naj bi uporabnik strežniku poslal neke podatke, ki se naknadno shranijo v podatkovno bazo.

   V ogrodju Laravel je za vsak URI, ki je javno dostopen, potrebno definirati pot ter dovoliti dostop do njega. Ko odjemalcu dovolimo dostop, lahko prek HTTP protokola do njega dostopa. V datoteki *app/Http/routes.php* so določene akcije, ki se zgodijo ob klicu. Lahko se kliče krmilnik ali pa se izvede preusmeritev na nek drug pogled. Če je akcija nedefinirana, nam strežnik odgovori z napako, ki ima kodo 500. Slika 2.10 prikazuje primer definiranja odgovorov na HTTP zahtevke.

```
/* GET REQUESTS */
Route::get('normPage', function(){

    return View::make('normPage');
});

Route::get('newsPage','NewsController@selectNews');

Route::get('userProfile','UserController@showProfile');
Route::post('userProfile','UserController@showProfile');

/* POST REQUESTS */
Route::post('/ajax/getWepAverage','AjaxController@getWepAverage');
Route::post('/ajax/getTournamentData','AjaxController@getTournamentData');
Route::post('/ajax/getTournamentDetail','AjaxController@getTournamentDetail');
```

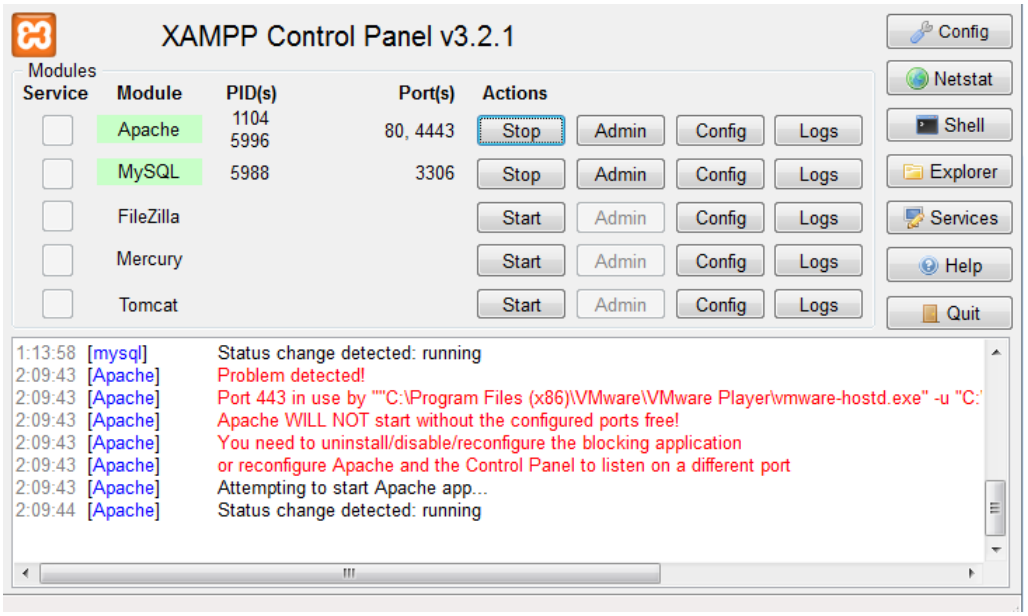
Slika 2.10: Vzorčni primer datoteke *routes.php* v ogrodju Laravel, ki definira odgovore na HTTP zahtevke.

4. **Blade predloge** Laravel uporablja Blade predloge za bolj preprosto in učinkovito pisanje Blade pogledov. Dovoljuje celo uporabo preproste PHP kode v pogledih. Vsi Blade pogledi se sestavijo v PHP kodo in so predpomnjeni, dokler se ne spremenijo, kar naredi aplikacijo hitrejšo in odzivnejšo. Vsi pogledi, ki uporabljajo Blade, morajo imeti končnico *blade.php*.

2.3 Orodja

2.3.1 XAMPP

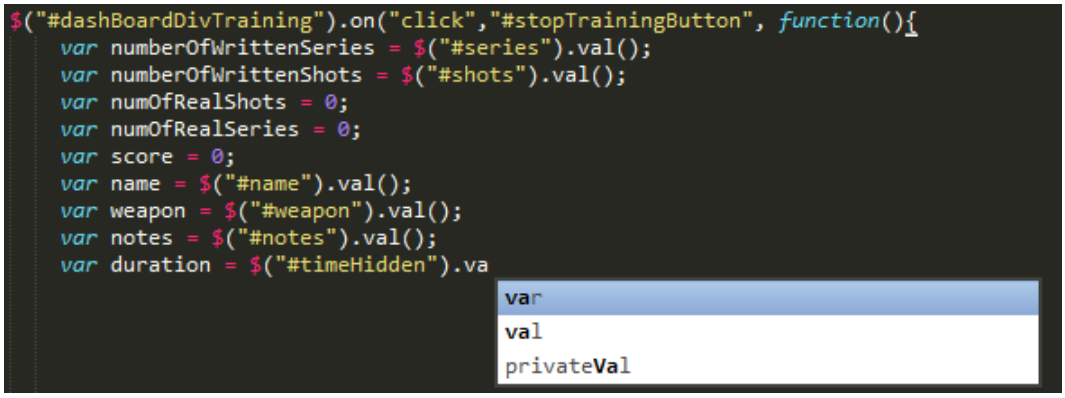
XAMPP je večplatformno prostodostopno odprtokodno orodje, ki ga razvijalci uporabljajo za vzpostavitev okolja za razvoj aplikacije [22]. Sama beseda XAMPP je akronim za večino orodij, ki jih aplikacija ponuja. Vsebuje Apache spletni strežnik, na katerem lahko razvijalec preizkuša, kako deluje strežniški del njegove aplikacije, MySQL sistem za upravljanje s podatkovnimi bazami, interpreterja za jezika PHP in Perl ter še mnoga druga orodja. Njegova največja prednost je enostavnost namestitve. Konfiguracija strežnikov je lahko precej zapletena zadeva, orodje XAMPP pa vse to naredi samo in je zato zelo uporabno za začetnike. Če želimo zagnati določeno storitev ali pa samo spremeniti nastavitve, lahko to naredimo iz nadzorne plošče, kot je prikazano na sliki 2.11.



Slika 2.11: Prikaz nadzorne plošče orodja XAMPP.

2.3.2 Sublime Text 2

Sublime Text je program za urejanje izvorne kode. Podpira veliko programskih in označevalnih jezikov in nam na več načinov pomaga pri razvoju aplikacij. Glavne prednosti so označevanje sintakse, simultano urejanje ter avtomatsko dopolnjevanje fraz. Ima tudi zelo napredeno upravljanje s paketi, saj lahko prek njega z lahkoto namestimo razne dodatne vtičnike, ki še niso nameščeni. Slika 2.12 prikazuje primer uporabe programa Sublime Text 2.



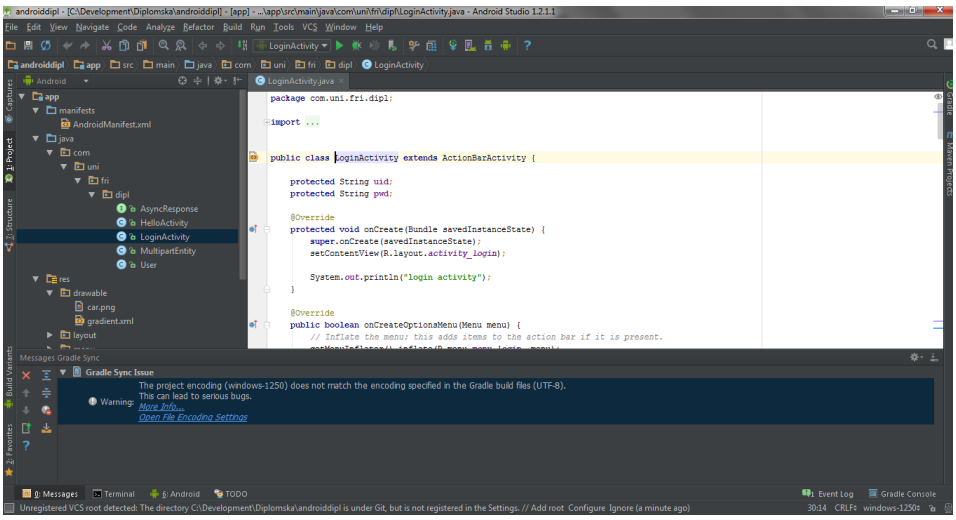
Slika 2.12: Prikaz označevanja sintakse in avtomatskega dopolnjevanja v programu Sublime Text 2.

2.3.3 Android Studio

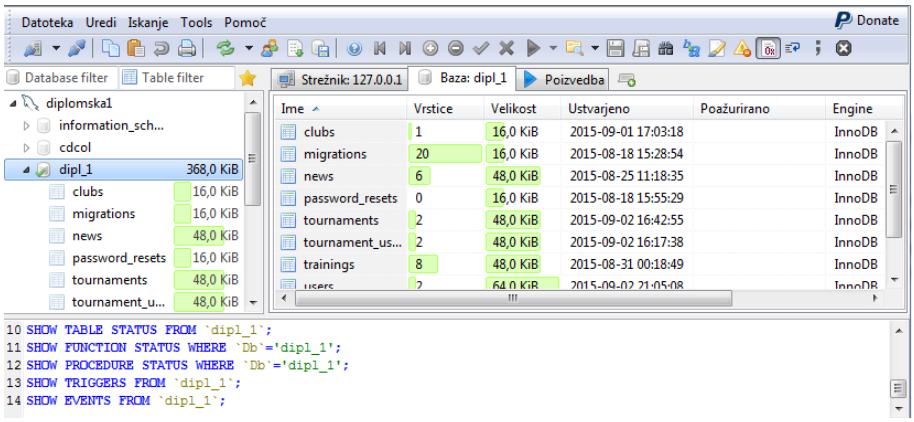
Android Studio je uradno razvojno okolje za razvijanje aplikacij za platformo Android [23]. Prva različica Android Studia je prišla na trg šele leta 2014 in je na voljo za operacijske sisteme Windows, Linux in Mac OS X. V primerjavi z Eclipse ADT, ki je bilo prejšnje uradno razvojno okolje, ponuja veliko izboljšav. Ima intuitivnejši in bolj pregleden vmesnik, boljše avtomatsko dopolnjevanje in še mnogo več. Grafični vmesnik Android Studia je prikazan na sliki 2.13.

2.3.4 HeidiSQL

HeidiSQL je orodje narejeno za spletne razvijalce, ki uporabljajo MySQL, Microsoft SQL ali PostgreSQL. Razvijalcu omogoča, da pregleduje in ureja podatke, ustvarja in ureja tabele, poglede ter procedure, poleg tega pa vsebuje tudi veliko drugih funkcionalnosti [24]. Vse te funkcionalnosti lahko razvijalec uporablja prek preprostega grafičnega vmesnika, ki je prikazan na sliki 2.14.



Slika 2.13: Prikaz grafičnega vmesnika Android Studia.



Slika 2.14: Prikaz grafičnega vmesnika orodja HeidiSQL.

Poglavje 3

Razvoj spletne aplikacije

Glavni vsebinski del našega diplomskega dela je bil ravno razvoj spletne aplikacije za strelska društva. Najprej smo pogledali že obstoječe rešitve in videli, da nobena od njih ne izpolnjuje minimalnih zahtev, ki jih uporabnik potrebuje pri tovrstni aplikaciji, torej bo naša aplikacija prva na tržišču. Zelo pomembna stvar pri razvoju spletne aplikacije je pogovor z nekom, ki bo to spletno aplikacijo redno uporabljal. Za to seveda obstaja veliko razlogov, glavni razlog pa je ta, da hočemo s tem zmanjšati uporabniško vrzel [25]. Uporabniška vrzel nastane takrat, kadar uporaba ne sledi nakupu, oziroma, ko naredimo preveč funkcionalnosti, ki jih potem nihče ne uporablja. Ta vrzel se lahko hitro izračuna kot razlika med številom funkcionalnosti, ki so bile kupljene in številom funkcionalnosti, ki se jih uporablja. Z manjšanjem vrzeli, se veča zadovoljstvo stranke in s tem tudi zaupanje v nas. Po pogovoru z nekaterimi posamezniki, ki se ukvarjajo s strelstvom, smo si zapisali cilje, ki naj bi jih naša spletna aplikacija izpolnila in prednosti, ki jih bo ponujala. Ko je bilo načrtovanja konec, smo se osredotočili na tehnologije, ki jih bomo uporabljali. Ker je še vedno največ spletnih strani postreženih na strežnikih, ki uporabljajo tehnologijo PHP [9], smo se odločili za to tehnologijo. Ker pa se je obširnega projekta težko lotiti brez ogrodja, smo si za izdelavo spletne aplikacije izbrali ogrodje Laravel, ki nam razvoj olajša, saj nam ponuja MVC arhitekturo, ki je pri razvoju novejših spletnih aplikacij

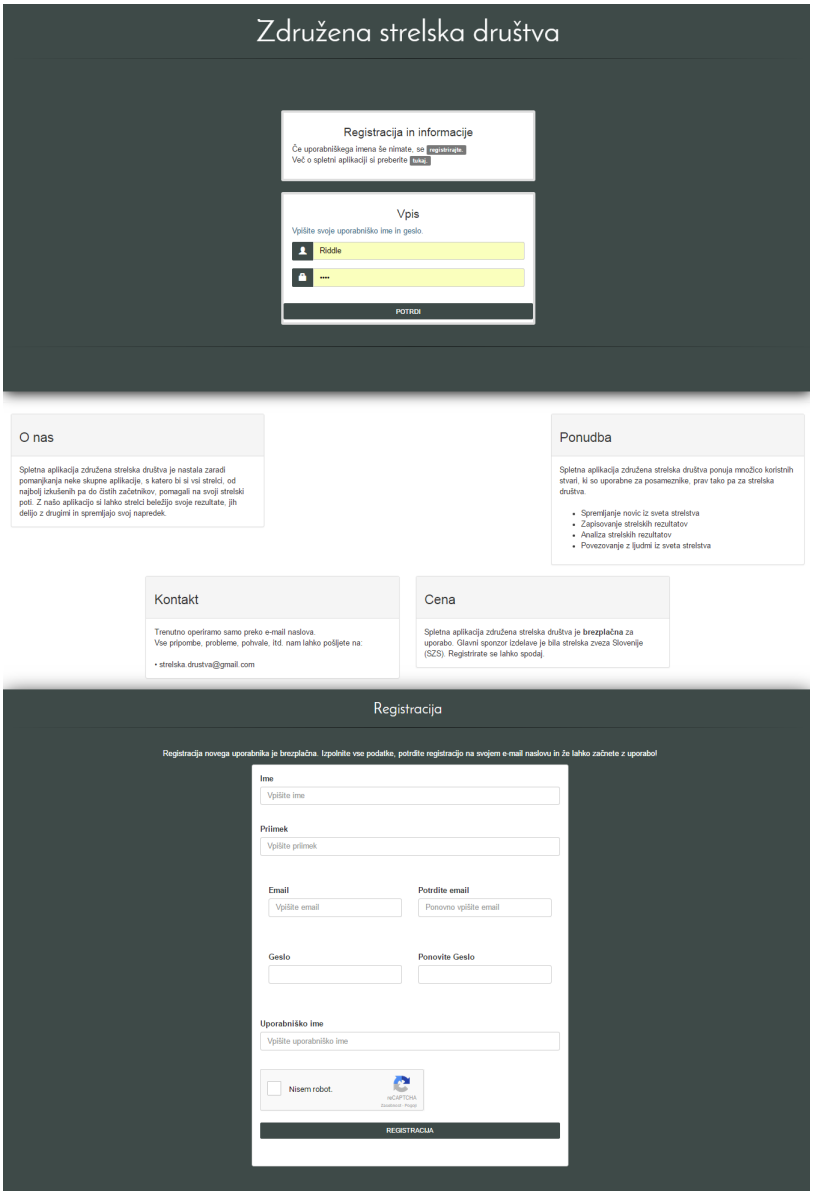
zelo popularna. Ker mora naša spletna aplikacija shranjevati podatke, smo za shranjevanje uporabili MySQL relacijsko podatkovno bazo.

3.1 Izdelava uporabniškega vmesnika

Ker razvijamo novo stvar na tržišču, pomeni, da še nimamo obstoječih uporabnikov in jih bo potrebno še pridobiti. Zato je izgled aplikacije še toliko bolj pomemben, saj je to prva stvar, ki jo nov uporabnik vidi. Potrebno se je tudi vprašati, za katero starostno skupino razvijamo aplikacijo. V našem primeru so ciljna skupina vsi ljudje, ki se ukvarjajo s strelstvom. To je precej široka starostna skupina, saj se strelske kategorije začnejo že s cicibani, ki so mlajši od 11 let, končajo pa s člani, ki so vseh starosti, od 21 let naprej. Kljub temu pa smo se osredotočili predvsem na strelsko kategorijo člani. Odločili smo se za izbiro "Jello" [26] oblikovanja, kjer je vsebina centrirana, ob strani pa imamo obrobo. Ena izmed glavnih prednosti tega oblikovanja je dober prikaz pri različnih ločljivostih.
Ker je razvijanje spletnega uporabniškega vmesnika lahko zelo zamudno delo, smo si pri tem pomagali z ogrodjem Bootstrap (opisano v poglavju 2.2.1). Ogrodje nam je zelo pomagalo pri izdelavi telesa strani, saj uporablja sistem mreže. Bootstrap ima vnaprej definirane razrede, kot sta *row* in *col*, ki razdelita telo na vrstice in stolpce. Največje število stolpcev, ki jih lahko določimo v eni vrstici, je 12. Ti stolpci in vrstice se znajo ob spremembi višine ali širine okna prestaviti tako, da se izgled strani še vedno ohrani, kar nam omogoča veliko večjo odzivnost in prilagodljivost različnim dimenzijam zaslonov.
Uporabniški vmesnik lahko razdelimo na tri samostojne enote, saj smo jih razvijali vsako posebej. Čeprav je ponavljanje eno izmed glavnih principov oblikovanja [27], smo se za izdelavo treh enot odločili zato, ker so ločene po namenu, ki mu služijo.

3.1.1 Prva stran

S terminom prva stran označujemo stran, na katero pride uporabnik, ko stran prvič obišče. Prva stran je prikazana na sliki 3.1. Če je uporabnik že vpisan v aplikacijo, potem se ta stran preskoči in prestavi uporabnika na naslednjo. Prva stran ima zelo preprost izgled in je sestavljena iz treh komponent. Prva komponenta se nam prikaže takoj, ko spletno stran odpremo, saj ima v CSS datoteki nastavljen atribut, da se razteza čez celotno višino in širino zaslona. Za barvo prve strani in hkrati barvo, ki se pojavlja čez celotno aplikacijo, smo si izbrali temno sivo barvo, ki ima vrednost RGB *(62,74,72)*. Na vrhu prve strani se pokaže ime naše strani "Združena strelska društva", pod tem pa se nahajata dve plošči bele barve (Slika 3.1, zgoraj). Na prvi plošči lahko najdemo povezavo do registracije in do informacij, na drugi plošči pa imamo obrazec za vpis. Druga komponenta prve strani se nahaja tik pod prvo,tu pa gre za komponento z informacijami (Slika 3.1, sredina). Ta komponenta se razteza čez polovico višine zaslona in vsebuje štiri plošče, na katerih se nahajajo informacije o spletni strani. Ima belo ozadje, ki se preliva v barvo prve komponente. Tretja komponenta (Slika 3.1, spodaj) je zelo podobna prvi in se nahaja tik pod drugo. Zavzema najmanj 100 odstotkov višine in širine zaslona in vsebuje registracijski obrazec. Barva ozadja je enaka kot pri prvi strani.



Slika 3.1: Prikaz grafičnega izgleda prve strani.

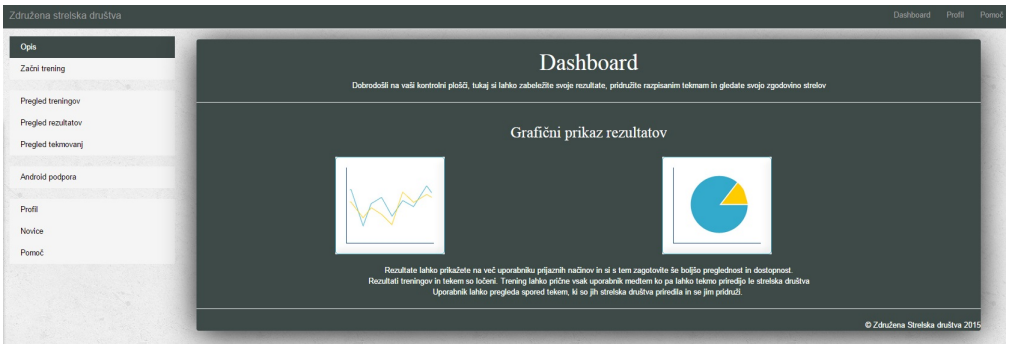
3.1.2 Stran z novicami

Do te strani lahko uporabnik pride s prve strani tako, da se uspešno vpiše. Na vrhu strani z novicami se nahaja glava, ki je enaka za vse strani, ki uporabljajo to predlogo. Stran z novicami je prikazana na sliki 3.2. Vseh teh strani je sedem. V glavi strani se nahaja navigacijska vrstica, ki vsebuje povezave do naslednjih strani. Nad njo se na levi strani nahaja ime naše aplikacije, na desni strani zaslona pa se nahaja polje, kjer je napisano uporabnikovo uporabniško ime. Pod glavo se nahajajo novice. Najprej je vrtiljak z novicami, kjer se v neskončni zanki premikajo tri novice. Pod vrtiljakom pa se nahajajo bolj obširne novice. Vsaka novica je objavljena v svoji plošči, vse te plošče pa so nameščene na razdelek za novice. Pod tem se nahajata še prostor za sponzorje in pa noga strani. Na nogi strani najdemo razne uporabne povezave povezane s spletno aplikacijo.

Slika 3.2: Prikaz grafičnega izgleda strani z novicami.

3.1.3 Nadzorna plošča

Nadzorna plošča, ki je prikazana na sliki 3.3, vsebuje za uporabnika najbolj uporabne funkcionalnosti, kot so beleženje rezultatov treningov, pregled treningov ter tekmovanj. Na prvi pogled ima precej osnoven izgled, vendar je tak izgled nastal z razlogom. Ravno ta del aplikacije bo uporabljalo največ uporabnikov, ki prihajajo iz vseh starostnih skupin, zato mora biti izgled minimalističen in prilagojen zelo različnim starostim.. Na vrhu imamo glavno vrstico, kjer imamo povezavi nazaj na stran z novicami ter na profil uporabnika. Stran je nato razdeljena na levi del, v katerem se nahaja navigacijska vrstica, kjer izbiramo med funkcionalnostmi, ki jih aplikacija ponuja, ter desni del, kjer je prikazana vsebina zavihkov. Ob kliku na določeno funkcionalnost se ta pokaže na predstavitveni plošči, ki se nahaja desno od navigacijske vrstice.



Slika 3.3: Prikaz nadzorne plošče.

3.1.4 Odzivnost uporabniškega vmesnika

Ker se v današnjih časih tehnologija seli na mobilne naprave, je potrebno tudi spletne aplikacije temu prilagoditi. Na srečo ima ogrodje Bootstrap zelo veliko elementov, ki se dobro obnašajo ob spremembi velikosti zaslona. Večina elementov se ob zmanjšanju strani postavi eden pod drugega, tako da ponovno tvorijo lepo oblikovano stran. Prva stran in nadzorna plošča naše aplikacije se dobro odzivata ob spremembi velikosti in se prilagajata. Na

nadzorni plošči smo uporabili stransko navigacijsko vrstico, ki se v primeru premajhnega zaslona skrije, in se, če jo uporabnik spet potrebuje, lahko ob kliku na gumb zopet razširi v vidno polje uporabnika. Tudi stran z novicami je pretežno prilagojena za uporabo na večini naprav, vendar ima nekaj zavihkov težave z manjšanjem zaslona, ki pa jih še nismo uspeli odpraviti. Slika 3.4 prikazuje prilagojen prikaz strani na napravah z manjšim zaslonom.

Slika 3.4: Prikaz nadzorne plošče ob spremembi širine in višine okna. Klik na puščico na vrhu odpre navigacijsko vrstico.

3.2 Povezava s strežnikom in njegova izdelava

Pri izdelavi spletnih aplikacij potrebujemo strežnik, ki bo odjemalcem stregel podatke. Za testiranje naše aplikacije smo morali izbrati nek strežnik, ki podpira uporabo jezika PHP. Dandanes ta jezik podpira že večina spletnih strežnikov. Ker pa bi izbira oddaljenega strežnika prinesla kar nekaj nevšečnosti, kot sta večja latenca in možnost izpada, smo se za razvoj odločili na lokalnem računalniku. Odločili smo se za uporabu orodja XAMPP, ki ponuja okolje, primerno za razvoj aplikacije, kot je naša. Na lokalni strežnik smo nato s pomočjo programa Composer naložili ogrodje Laravel in začeli z razvojem.

3.2.1 Povezava s strežnikom

Pregledati smo morali, kako se bodo uporabniki lahko povezovali na naš strežnik. Ker ogrodje Laravel temelji na arhitekturi REST, ki je opisana v poglavju 2, smo morali najprej določiti, katere metode bomo na našem strežniku dovolili. Odločili smo se za uporabo GET in POST metod, s katerimi lahko prek protokola HTTP pridobimo podatke, ki jih obdelamo, ali pa shranimo v bazo. Razlika med GET in POST zahtevkom je ta, da metoda GET prenaša parametre v naslovu resursa, POST pa v HTTP zahtevku, kar naredi POST zahtevek bolj varen, saj se podatki ne hranijo, kar je posebno priročno za pošiljanje gesel in drugih občutljivih podatkov. Torej do našega RESTful APIja lahko uporabnik dostopa z vpisovanjem URIjev v spletnem brskalniku. Na nekaterih mestih v aplikaciji, kot na primer ob registraciji ali vpisu, se ob pritisku na gumb sproži forma, ki sama pošlje podatke na točno določen URI na našem strežnik, ki nato podatke obdela in vrne ustrezno vsebino.

Poleg vseh teh načinov pa naša aplikacija do strežnika dostopa še s pomočjo tehnologij AJAX. Gre za drugačen način dostopa kot pri prejšnjem primeru, saj gre za asinhron dostop do strežnika. To pomeni, da spletna aplikacija lahko strežniku pošlje podatke v ozadju, brez da bi se uporabnik zavedal,

da v ozadju to poteka. Razlika pa je tudi v tem, da se, ko odjemalec dobi podatke iz strežnika, spletna stran ne naloži ponovno v celoti, pač pa se samo posodobi nek del spletne strani. V naši aplikaciji je AJAX uporabljen praktično ob vsakem kliku, ki ga uporabnik naredi v svoji nadzorni plošči. Ko uporabnik pritisne na gumb za začetek treninga, se pošlje zahtevek na strežnik, ki odgovori z ustreznim pogledom, ki je v tem primeru pogled za izris treninga. Slika 3.5 prikazuje AJAX klic, ki se izvede ob zahtevi za shranitev treninga. AJAX klic lahko na spletni strani poženemo samo s pomočjo JavaScripta, ki je dinamičen del spletne strani. Za AJAX klice uporabljamo knjižnico jQuery, ki celoten postopek AJAX klicev zelo poenostavi. Podatki se prek AJAX klicev pošiljajo v formatu XML ali JSON, strežnik pa jih nato razčleni in uporabi pri izvedbi procedur.

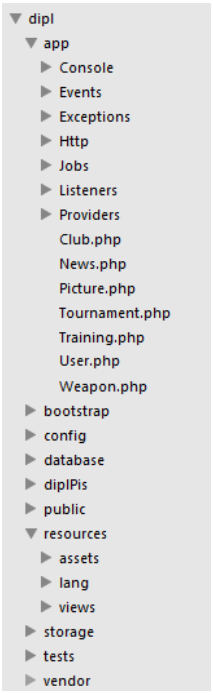
```
$.ajaxSetup({
    headers: {
            'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
        },
});
$.post('ajax/saveTraining',{name:name,numOfShots:numOfRealShots,numOfSeries:numOfRealSeries,
  score:score,notes:notes,weapon:weapon,duration:duration}, function(data) {
      if(data.success == 1){
          $("#trainingSuccess").text(data.message);
          $("#trainingSuccess").css('visibility', 'visible');
      }else{
          $("#trainingFail").text(data.message);
          $("#trainingFail").css('visibility', 'visible');
      }
  });
```

Slika 3.5: Prikaz AJAX klica, ki ga pošljemo, ko strelec shrani trening.

3.2.2 Implementacija strežnika

Do sedaj smo pogledali samo komunikacijo med odjemalcem in strežnikom, nič pa še nismo povedali, kaj točno se s podatki zgodi, ko pridejo do strežnika. Način obdelave podatkov na strežniku je odvisen od strežnika ter programov, ki so nanj nameščeni. V našem primeru uporabljamo ogrodje Laravel, ki podatke obdeluje na svoj način. Ko pošljemo zahtevo do strežnika, jo tam prestreže najnižji programski sloj, in sicer usmerjevalnik. V naši imeniški strukturi, ki je predstavljena na sliki 3.6, lahko najdemo usmerjevalnik v

mapi *app/Http/routes.php*. V usmerjevalniku so definirane tiste zahteve, ki jih naša poslovna logika implementira. Če poslana zahteva ne ustreza nobenemu pogoju v našem usmerjevalniku, bo strežnik vrnil nazaj napako *NotFoundHttpException*, če pa zahteva pogoju ustreza, se izvede, kar je za to zahtevo definirano. Obstaja veliko načinov, kako lahko usmerjevalnik zahtevo obdela. Najbolj osnoven način je, da kar iz usmerjevalnika vrnemo odgovor ali pa enega izmed pogledov. Če pa ima zahteva kake podatke, ki jih želimo še dodatno obdelati, potem se lahko dvignemo nivo višje, na nivo krmilnika. Ena izmed posebnosti ogrodja je ta, da vsebuje tudi srednji nivo. To je nivo, ki se nahaja med usmerjevalnikom in krmilnikom. Srednji nivo se v imeniški strukturi ogrodja Laravel nahaja v mapi *app/Http/Middleware*.



Slika 3.6: Laravel imeniška struktura

Odgovoren je za filtriranje HTTP prometa, ki prihaja v našo aplikacijo. Uporabi se ga lahko za marsikaj, kot je na primer beleženje vseh zahtev. V našem primeru smo ta nivo uporabljali za avtentikacijo uporabnika, kjer preverjamo, ali je uporabnik že avtenticiran, ter za preprečevanje CSRF napadov.
Nato pridemo na nivo krmilnikov. Namesto, da definiramo vse svoje akcije že kar v usmerjevalniku, je lepše, če delovanje organiziramo tako, da se poslovna logika izvede v krmilnikih. Krmilniki lahko grupirajo HTTP zahteve v smiselne skupine in jih na enem mestu obdelajo. Krmilniki so običajno shranjeni v mapi *app/Http/Controllers*. Krmilnik je nekakšna povezava med pogledi in modeli, saj lahko prek modelov dostopa do podatkovne baze in na podlagi teh podatkov spreminja poglede. V naši aplikaciji smo definirali pet krmilnikov, ki smo jih razdelili na logične sklope:

- AjaxController

  To je krmilnik, ki obdela največ zahtevkov, saj obdeluje vse AJAX

zahtevke, ki jih odjemalec pošlje na strežnik. V to so vključeni mnogi obrazci, pregled zasedenosti imena, posodobitev profila in mnogi drugi.

- DashboardController
  V tem krmilniku se nahajajo zahtevki, ki jih odjemalec pošlje, ko uporablja svojo nadzorno ploščo. Vse, kar je povezano s nadzorno ploščo, se obdela v tem krmilniku, vključno z AJAX zahtevki.

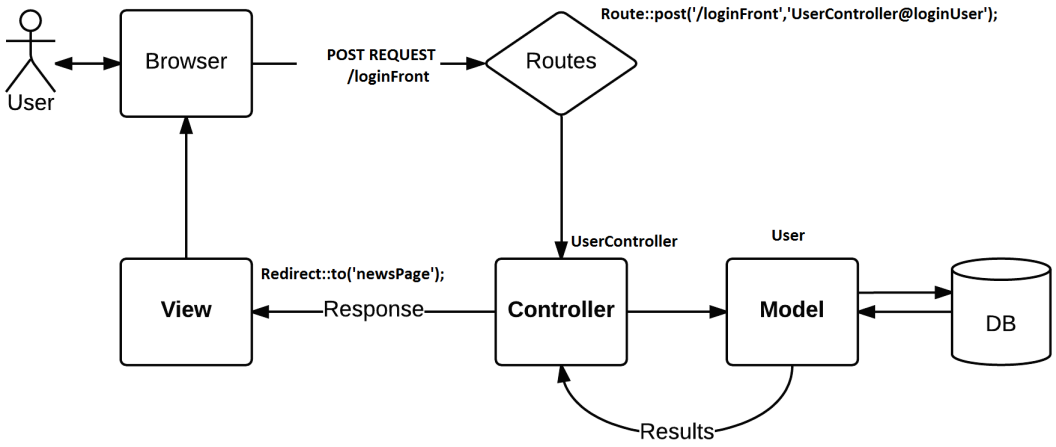
- NewsController
  Ta krmilnik obdeluje vse zahtevke, ki spreminjajo novice.

- UserController
  S tem krmilnikom se spreminja vse nastavitve uporabnika. Od njegove registracije in vpisa, do spremembe privatnosti.

- HomeController
  To je avtomatsko generiran krmilnik, ki obdela vse ostale HTTP zahtevke, ki ne spadajo v nobenega od zgoraj naštetih. Sem spada tudi obdelava podatkov, ki jih dobimo prek naše Android aplikacije, ki bo opisana kasneje.

Taka razdelitev krmilnikov nam močno olajša programiranje, saj razvijalec ves čas ve, kje se nahaja določena logika, kar močno olajša razvoj.

Kot vidimo na sliki 3.7, naš krmilnik po končanem delovanju uporabniku vrne odgovor. Ti odgovori so lahko neobičajni, kot npr. JSON ali PDF, ponavadi pa krmilnik vrne uporabniku pogled, ki ga potem spletni brskalnik izriše na zaslonu. Lahko gre za celoten PHP dokument ali pa, kot pri AJAX klicih, le nek del HTML kode, ki se nato s pomočjo JavaScripta vstavi, kamor je to potrebno. Pogledi so v Laravel ogrodju shranjeni v mapi *app/resources/views*. V naši aplikaciji smo definirali 12 različnih osnovnih pogledov, ki jih uporabljamo čez celotno aplikacijo. To so pogledi, kot so **dashboard.blade.php**, ki nam definira uporabnikovo nadzorno ploščo, **userProfile.blade.php**, ki predstavlja pogled profila uporabnika ter **index.blade.php**, s katerim prikažemo odjemalcu prvo stran. Kot smo že



Slika 3.7: Prikaz MVC arhitekture ogrodja Laravel na primeru uspešnega vpisa uporabnika, povzeto po [28].

na kratko predstavili v poglavju 2, končnica *.blade.php* predstavlja datoteko, ki uporablja Blade predloge. Te predloge so zelo uporabne, kadar želimo uporabiti podatke iz krmilnika v pogledu, saj jih lahko s preprosto sintakso '{{ ... }}' uporabimo v pogledu. Če jih želimo uporabiti v pogledu, moramo seveda ob klicu prikaza pogleda v krmilniku povedati še, katere podatke lahko pogled uporabi. To naredimo tako, kot je prikazano na sliki 3.8.

```
<p>Dodajte orožja:</p>
<label for="wepsList">Orožja</label>
<select class="form-control" userId="{{$user->id}}" id="wepsList" name="weapon">
    @foreach ($weapons as $weapon)
        <option value="{{ $weapon->id}}">{{ $weapon->caliber}} {{ $weapon->name}} {{ $weapon->type}}</option>
    @endforeach
</select>
```

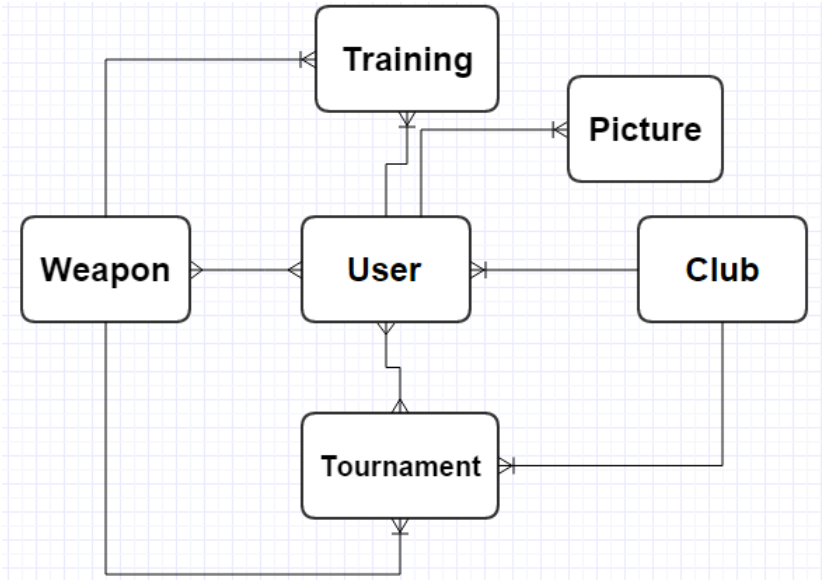
Slika 3.8: Prikaz uporabe podatkov iz krmilnika v pogledu na primeru polnjenja spustnega menija.

Poleg osnovnih pogledov pa smo definirali še mnoge druge, vsi pogledi, ki jih vrnemo prek AJAX klicev, pa se nahajajo v mapi *app/resources/dashDiv*. Prav tako smo definirali dva vključujoča pogleda, katera vključimo v skoraj vsak osnovni pogled, in sicer **header.blade.php** ter **footer.blade.php**, ki

vsebujeta kodo za glavo in nogo spletne aplikacije. Definiran je tudi pogled, ki ima čisto drugačno vlogo, to je **emails.welcome.php**, ki pa vsebuje kodo potrditvene elektronske pošte, ki se jo uporabniku pošlje ob registraciji v aplikacijo.

3.3 Povezava s podatkovno bazo

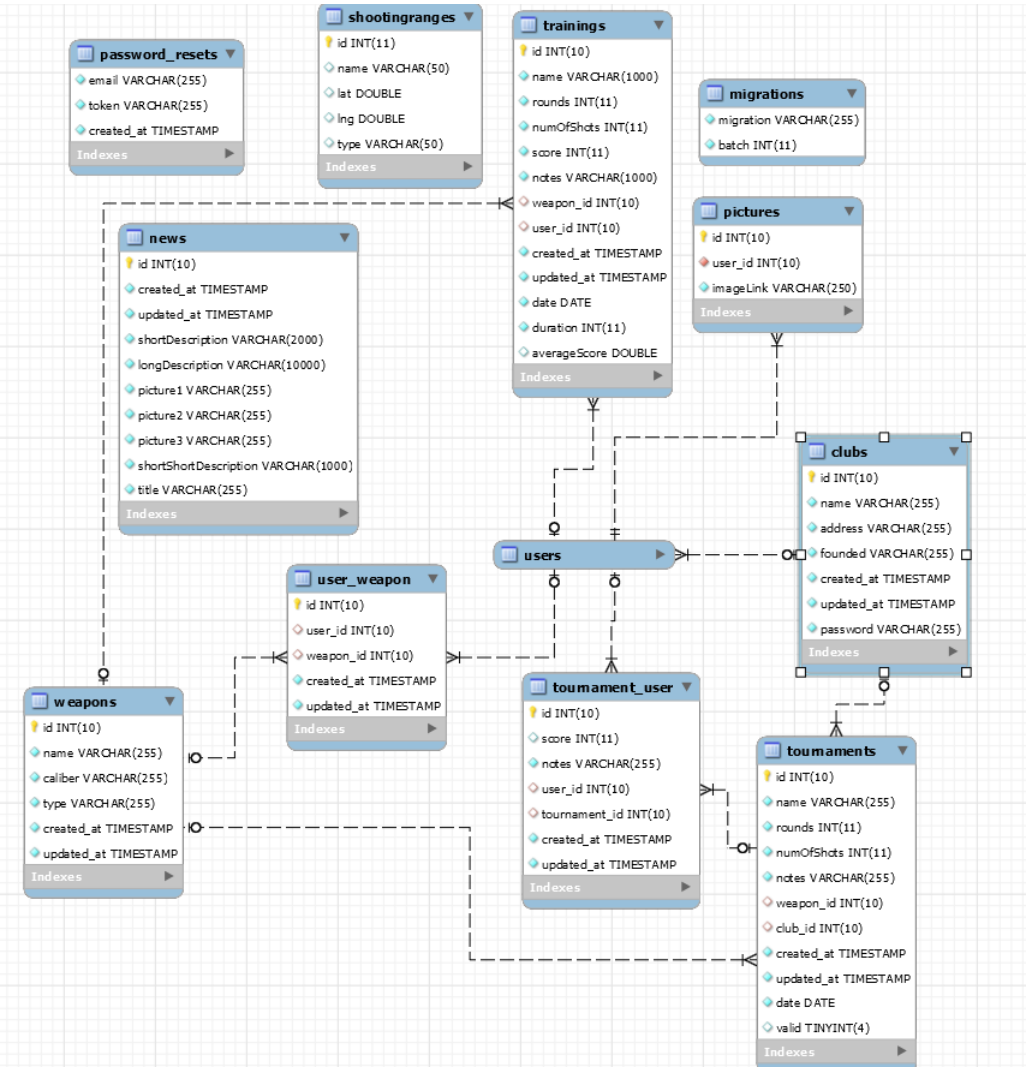
Še zadnja dva gradnika Laravel MVC arhitekture, predstavljene na sliki 3.7, ki ju nismo opisali, so modeli in podatkovna baza. Najprej smo si zamislili konceptualni model naše podatkovne baze. Konceptualni model nam na višjem nivoju definira naše entitete [29]. To pomeni, da vemo, kaj naše entitete so in kakšne so povezave med njimi. Konceptualni model je predstavljen na sliki 3.9. Uporabnik je član enega strelskega kluba in ima v lasti več orožij. Uporabnik lahko prične več treningov in se pridruži večim turnirjem. Prav tako lahko na treningih slika svoje tarče in si jih shrani na strežnik. Trening ima lahko samo enega uporabnika, ki trenira z enim orožjem. Na turnir se lahko pridruži več uporabnikov. Izvaja se z enim orožjem, ki je enak za vse uporabnike. Turnir vedno priredi le en klub. Strelski klub ima lahko enega ali več pridruženih članov, prav tako pa lahko priredi enega ali več turnirjev, na katerega se lahko pridružijo člani kateregakoli kluba.



Slika 3.9: Konceptualni model podatkovne baze.

Po razvoju konceptualnega modela smo razvili še logični in fizični model, ki smo ga na koncu kreirali v relacijski podatkovni bazi MySQL. Odpravili

smo vse več-na-več *(angl. 'many-to-many')* povezave, ki so obstajale v bazi, in kreirali vmesne tabele. Po potrebi smo ob razvoju ustvarili še kakšno tabelo, ki smo jo pri načrtovanju pozabili. Slika 3.10 prikazuje končno implementacijo podatkovne baze.



Slika 3.10: Končna implementacija podatkovne baze.

Ker smo v spletni aplikaciji uporabljali arhitekturo MVC z ORM, je bil za uspešno delovanje razviti še model v ogrodju Laravel. V imeniški strukturi ogrodja se modeli nahajajo v mapi *app*. S pomočjo modelov se znebimo

pisanja dolgih in zahtevnih SQL stavkov, saj ogrodje vse to naredi za nas, potrebno je le, da v kodi kreiramo objekt modela in nato nad njim izvajamo operacije, kot nad navadnimi objekti jezika. Konvencija ogrodja določa, da naj bi bilo ime objekta edninski samostalnik tabele, ki jo predstavlja v podatkovni bazi, pisan z veliko začetnico. Modeli naj bi predstavljali le osnovne tabele, kot so uporabnik, turnir ali orožje. Na sliki 3.11 je prikazan model uporabnika. Do tabele, ki jo ustvarimo ob razbitju več-na-več povezav, lahko dostopamo prek obeh tabel, ki sta nanjo povezani. Model pa ne predstavlja samo objekta, ki se preslika v tabelo v podatkovni bazi, saj ima lahko še veliko drugih opcij. Lep primer je določitev validacijskih pravil, ki se morajo vedno upoštevati, preden se objekt shrani v podatkovno bazo.

```php
class User extends Model implements AuthenticatableContract, CanResetPasswordContract
{
    use Authenticatable, CanResetPassword;
    protected $table = 'users';
    protected $fillable = ['name', 'email', 'password','club_id'];
    protected $hidden = ['password', 'remember_token'];
    public static $rules = array(
        'name'             => 'required',
        'surname'          => 'required',
        'email'            => 'required|email|unique:users',
        'password'         => 'required',
        'password_confirm' => 'required|same:password',
        'username'         => 'required|unique:users'
    );
    public static $ruleMsg = [
            'name.required'      => 'Uporabniško ime je treba izpolniti',
            'surname.required'   => 'Geslo ime je treba izpolniti',
            'email.unique:users' => 'Email naslov je že registriran na naši strani',
            'email.email'        => 'Napačen format email naslova.',
            'email.required'     => 'Email je potrebno izpolniti',
            'password.required'  => 'Potrebno je vnesti geslo',
            'password_confirm.required'  => 'Potrebno je vnesti ponovno vnesti enako geslo',
            'username.required'  => 'Potrebno je vnesti veljavno uporabniško ime.',
            'username.unique:users' => 'Uporabniško ime je že zasedeno',
        ];
    public function club() {
        // this matches the Eloquent model
        return $this->belongsTo('App\Club');
    }
     public function weapons()
    {
        return $this->belongsToMany('App\Weapon')->withTimestamps();
    }
    public function tournaments()
    {
        return $this->belongsToMany('App\Tournament')->withPivot('score', 'notes')->withTimestamps();
    }
}
```

Slika 3.11: Primer modela uporabnika v ogrodju Laravel.

Poglavje 4

Razvoj Android aplikacije

Poleg spletne aplikacije smo, kot del diplomske naloge razvili, tudi mobilno aplikacijo za operacijski sistem Android. Kljub vsem funkcionalnostim, ki jih spletna aplikacija ponuja, smo po pogovoru s strelci ugotovili, da jih večina strelskih rezultatov raje sploh ne bi zapisovala, oziroma bi jih raje zapisovali po treningu. Kot razlog so navedli, da jih zapisovanje rezultatov med treningi moti, ker izgubijo pozornost. Zato smo se odločili razviti tudi mobilno aplikacijo, ki bi strelcem ta del treninga olajšala. Za izdelavo smo se odločili kljub temu, da je zapisovanje strelskih rezultatov v nadzorni plošči spletne aplikacije prilagojeno za uporabo na mobilnih telefonih. Na spletni aplikaciji si morajo strelci sproti zapisovati rezultate, s pomočjo Android aplikacije pa strelcu ponudimo alternativno možnost. S pomočjo Android aplikacije lahko strelec po končanem treningu slika svoje tarče in te slike naloži na naš strežnik. Tam se nato slike shranijo v podatkovno bazo tako, da je jasno, katera slika pripada kateremu uporabniku. Kasneje, ko ima strelec čas za vpis svojih rezultatov, si lahko, na posebnem zavihku *Android podpora* v nadzorni plošči spletne aplikacije, te slike pogleda in rezultate vpiše naknadno. S tem strelcem olajšamo strelske treninge, saj jim ni treba skrbeti za sprotno vpisovanje rezultatov. Vse kar morajo narediti je vzeti svoj telefon in slikati svojo tarčo.

Dandanes že večina ljudi uporablja pametne telefone. Pri izdelavi mobilne

aplikacije se je potrebno vprašati za kater operacijski sistem jo bomo razvijali. Mi smo se odločili razvijati za operacijski sistem Android, saj ima daleč največji delež na trgu [30]. Za razvijanje pa smo uporabili uradno razvojno orodje podjetja Google, Android Studio, ki omogoča razvoj v programskem jeziku Java, v katerem je tudi razvita večina Android aplikacij.

4.1 Uporabniški vmesnik

Kot smo že omenili v podpoglavju 3.1, je naša ciljna starostna skupina zelo široka, zato je potrebno narediti predvsem enostavno oblikovanje, ki ga bodo uporabniki znali uporabljati. Čeprav je Android aplikacija ločena od spletne aplikacije in jo lahko uporabnik uporablja samostojno, smo se odločili upoštevati oblikovalski princip ponavljanja. Za uporabniški vmesnik Android aplikacije smo izbrali enako temno sivo barvo, kot smo jo uporabili za izdelavo spletne aplikacije. Za aplikacijo smo uporabili tudi enako ime in enak logo Strelske zveze Slovenije. Android aplikacija ima dva pogleda:

- Pogled prijave
  Prvi pogled je pogled prijave. Ta pogled je potreben, da se uporabnik avtenticira preden začne nalagati slike na strežnik. Izgled je zelo preprost, kot lahko vidimo na sliki 4.1.

  Na vrhu je napisano ime aplikacije *"Združena strelska društva"*. Pod tem se nahaja obrazec za vpis, ki vsebuje samo dve vnosni polji, uporabniško ime in geslo, ter gumb za vpis. Ko ima uporabnik fokus na enem izmed vpisnih oken, se spodnji del tega okna obarva s turkizno barvo. Na dnu se nahaja logo Strelske zveze Slovenije, ki je enak kot tisti, ki ga uporabljamo v spletni aplikaciji.

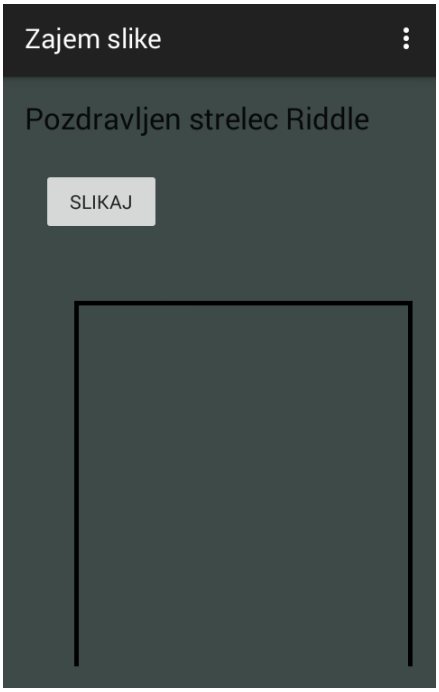
- Pogled zajemanja fotografij
  V tem pogledu lahko uporabnik zajema fotografije in jih shrani na strežnik. Na vrhu je izpisano ime pogleda *Zajem slike*, pod njim pa pozdrav uporabniku. Pod tem ima aplikacija en gumb, s katerim se

Slika 4.1: Pogled prijave v Android aplikaciji.

prižge kamera, tako da lahko uporabnik zajame sliko, in pa prostor, kjer se, po končanem zajemanju slike, uporabniku prikaže slika, ki jo je naredil. Slika 4.2 prikazuje pogled zajemanja fotografij.

4.2 Povezava s strežnikom

V Android aplikaciji smo naredili dve aktivnosti, **LoginActivity** in **HelloActivity**. Aktivnosti služita različnim namenom, vendar pa imata nekaj skupnih stvari. Obe se povezujeta na isti strežnik. Strežnik, do katerega se povezujeta, je strežnik, na katerem je tudi naša spletna aplikacija. Za testiranje smo uporabili lokalno omrežje. Da pa smo testiranje naredili bolj realno, smo uporabili dva računalnika, enega na katerem smo imeli lokalni strežnik in drugega za razvoj aplikacije, ter en mobilni telefon z Android operacijskim sistemom. Lokalni strežnik smo namestili s pomočjo ogrodja XAMPP. Ker je



Slika 4.2: Pogled zajemanja fotografij v Android aplikaciji.

bila na strežnik nameščena RESTful aplikacija, narejena z ogrodjem Laravel, smo na strežniku za potrebe Android aplikacije izpostavili dve storitvi. Obe storitvi sta dostopni prek HTTP protokola in njegovega POST zahtevka, na posebnem URIju:

- **LoginActivity**
  Ta aktivnost skrbi za pravilen vpis uporabnika v aplikacijo. Aktivnost se s strežnikom povezuje prek POST zahtevka na URIju *androidLogin*. Ob kliku na gumb *PRIJAVA* se na sistemu Android ustvari novo asinhrono opravilo (razred *AsyncTask)*, ki se v ozadju pošlje na strežnik. Ker je opravilo asinhrono, uporabnik niti ne ve, da se v ozadju kaj dogaja. V HTTP zahtevek se vstavita uporabniško ime in geslo, kot je prikazano na sliki 4.3. Ko zahtevek prispe do strežnika, ga ta preusmeri do pravilne funkcije v krmnilniku *HomeController*. V krmilniku se najprej izvede avtentikacija uporabnika. Če avtentikacija uspe, se Android aplikaciji pošlje nazaj JSON odgovor, ki vsebuje uporabniško

ime uporabnika. V nasprotnem primeru se pošlje nazaj prazen odgovor.

Android aplikacija sprejme odgovor s pomočjo metode *onPostExecute*. V tej metodi se pregleda strežnikov odgovor. Če je odgovor prazen, se v pojavno okno vpiše, da vpis ni uspel. Če pa vpis v aplikacijo uspe, aplikacija pokliče novo aktivnost. Temu klicu aktivnosti doda, kot argument, še uporabniško ime uporabnika.

```
StringEntity stringEntity = null;
try {
    stringEntity = new StringEntity("{ \"username\" : \"" + uid + "\", \"password\" : \"" + pwd + "\" }");
} catch (UnsupportedEncodingException e) {
    e.printStackTrace();
}

HttpClient httpclient = new DefaultHttpClient();
HttpPost httppost = null;
try {
    httppost = new HttpPost(new java.net.URI(params[0].toString()));
    httppost.setHeader("Content-type", "application/json");
    httppost.setEntity(stringEntity);
} catch (URISyntaxException e) {
    e.printStackTrace();
}
```

Slika 4.3: Izsek kode za pošiljanje HTTP zahtevkov v jeziku Java.

- **HelloActivity**

  Ta aktivnost skrbi za zajemanje slik in njihovo shranjevanje na strežnik. Tudi ta aktivnost se s strežnikom poveže prek POST zahtevka. POST zahtevek v Android aplikaciji naredimo s pomočjo Apache HTTP komponent [31]. Ko uporabnik klikne na gumb *SLIKAJ*, se pokliče Android aplikacija za kamero, s pomočjo katere zajamemo sliko. Ko nam ta aplikacija vrne sliko, se slika najprej pomanjša in kompresira, da jo lahko nastavimo v pogledu, nato se kliče funkcija za nalaganje slike na strežnik. Sliko nato s pomočjo razreda *ByteArrayOutputStream* pošljemo na strežnik. URI za sprejem zahtevka je *androidUpload*. Kateremu uporabniku slika pripada, ugotovimo s pomočjo imena slike, ki se v Android aplikaciji nastavi kot *"< Username > − < TimeInMiliseconds >.jpg"*. Nato sliko shranimo v mapo *public/img/uploads*. Slike vseh uporabnikov se nahajajo v tej mapi. Zato, da vemo, katera slika pripada kateremu uporabniku, se v podatkovno

bazo v tabelo slike shrani ime slike ter njej pripadajoči identifikator uporabnika. Uporabnik lahko shranjene slike pogleda v nadzorni plošči spletne aplikacije. Slika 4.4 prikazuje funkcijo napisano v jeziku PHP, ki na strežniku skrbi za shranjevanje slik na strežnik in njihovih imen v podatkovno bazo.

```php
public function androidUpload(){
    if(isset($_FILES['myFile'])){
        $filename = $_FILES['myFile']['name'];
        move_uploaded_file($_FILES['myFile']['tmp_name'], "uploads/" . $_FILES['myFile']['name']);
        $userExp = explode('-',$filename);
        $username = $userExp[0];
        $user = User::where('username','=',$username)->first();

        $picture = new Picture;
        $picture->user_id = $user->id;
        $picture->imageLink = $filename;
        $saved = $picture->save();
        if($saved){
            return "Slika shranjena";
        }else{
            return "Napaka. Slika ni bila shranjena.";
        }
    }else{
        return "No file sent";
    }
}
```

Slika 4.4: Funkcija v krmilniku, ki skrbi za shranjevanje slik na strežnik.

Poglavje 5

Pregled delovanja spletne in Android aplikacije

V prejšnjih poglavjih je bila opisana implementacija spletne strani na strežniku. V tem poglavju pa si bomo aplikacijo pogledali s strani uporabnikov. Poglavje je ločeno na tri dele. V prvem delu so predstavljene funkcionalnosti, ki jih lahko uporablja navaden uporabnik. V drugem delu se osredotočimo na strelska društva, ki imajo v spletni aplikaciji pomembno vlogo, saj lahko za razliko od uporabnika, organizirajo tudi turnirje. Ker imajo uporabniki in strelska društva veliko skupnih funkcionalnosti, bomo v tem poglavju predstavili le tiste, ki so za strelska društva unikatne. V tretjem delu pa si bomo pogledali, kaj lahko uporabnik počne z Android aplikacijo, ki je bila narejena kot dodatna pomoč strelcem pri njihovih treningih.

5.1 Funkcionalnosti uporabnika

Med uporabnike štejemo vse strelce, ki spletne aplikacije ne uporabljajo kot administrator strelskega društva, ampak želijo izkoristiti uporabniške funkcionalnosti naše aplikacije. Uporabnike v naši spletni aplikaciji ločimo po uporabniških imenih. Kljub temu, da mora uporabnik ob registraciji podati svoje ime, priimek in elektronsko pošto, smo se za uporabniška imena

odločili, ker nočemo razkrivati privatnih podatkov uporabnikov.
Poglavje bo razdeljeno glede na podstrani, ki jih uporabnik lahko obišče.

5.1.1 dipl/public/

To je prva stran na katero vsak nov uporabnik vstopi. Predstavljena je z **index.blade.php** in predstavlja vstopno točko do drugih podstrani. Če uporabnik še ni registriran, ima nad oknom za vpis, okno z informacijami. Tam ima uporabnik možnost klika enega izmed dveh gumbov. Gumb *registrirajte*, bo z animacijo uporabnika premaknil na dno strani, kjer se nahaja forma za registracijo. S klikom na gumb *tukaj* pa bo uporabnik premaknjen na predel med območjem za vpis in območjem za registracijo. Tam si lahko uporabnik prebere različne stvari, kot je na primer ponudba naše aplikacije in pa naš kontakt. Če pa je uporabnik v spletno aplikacijo že registriran, se lahko vpiše na oknu za vpis. Vpisati mora svoje uporabniško ime in geslo. V primeru, da vpiše pravilno ime in geslo, bo preusmerjen na stran z novicami. Če pa mu vpis ne bo uspel, kar pomeni, da se je zmotil pri uporabniškem imenu ali geslu, se mu bo nad vpisnim poljem za uporabniško ime prikazalo obvestilo o napačnem vnosu podatkov, kar je prikazano na sliki 5.1.

Slika 5.1: Primer vpisa z napačnimi podatki.

Uporabnik se lahko registrira na dnu strani, kjer se nahaja forma za

registracijo. Pri registraciji mora uporabnik vpisati svoje ime, priimek, elektronsko pošto, geslo in uporabniško ime. Vsa polja se na strani odjemalca pregledajo z regularnimi izrazi. V JavaScript datoteki za regularne izraze imajo vsa polja definirano akcijo za preverjanje. Ko se zamegli fokus na poljih ime in priimek, se preveri ali vsebujeta vsaj dve črki in če vsebujeta samo črke. Na poljih za vnos elektronske pošte se preverja ali je format vpisane elektronske pošte ustrezen, preverja pa se tudi, ali sta si obe pošti enaki. Če je ena pošta vpisana, druga pa ne, se okence obarva rdeče, spodaj pa se uporabniku pojavi sporočilo z napako. Podobno se preverja polji za geslo. Edina omejitev gesla je ta, da mora biti dajše od šest znakov. Zopet preverjamo, ali se gesli ujemata. Uporabniško ime ima za seboj AJAX podporo. Ko uporabnik vpiše svoje ime in se mu okno za vpis zamegli, se na strežnik pošlje zahteva po preverbi imena. Če ime v bazi že obstaja, se okence obarva rdeče in uporabnik mora vpisati drugo ime. Če pa ime še ni zasedeno, potem to pomeni, da je uporabniško ime prosto. V registracijski obrazec je vgrajena tudi reCAPTCHA. To je sistem podjetja Google, ki z določenim testom ugotovi, ali je uporabnik aplikacije človek ali robot. Ponavadi gre za zaznavo predmetov ali pa izpis številk, ki se prikažejo na zaslonu. Ko uporabnik izpolni reCAPTCHO, se prek AJAXa pošlje zahteva na strežnik podjetja Google, ki nazaj odgovori, ali je bil test rešen pravilno ali ne. Po opravljeni uspešni registraciji, se uporabnik še ne more takoj vpisati. Najprej mora potrditi elektronsko pošto, ki jo je prejel. Slika 5.2 prikazuje primer potrditvene elektronske pošte. Ta vsebuje povezavo do strani, kjer uporabnik aktivira svoj račun. Po aktivaciji računa se uporabnik lahko vpiše.

Slika 5.2: Primer potrditvene elektronske pošte.

5.1.2 dipl/public/newsPage

Ko se uporabnik uspešno vpiše, ga strežnik preusmeri na stran z novicam, ki je prikazana na sliki 3.2. Na vrhu strani se nahaja navigacijska vrstica, s katero se lahko uporabnik premika po straneh. V desnem zgornjem kotu, ob logotipu Slovenske strelske zveze, se nahaja gumb, na katerem piše uporabnikovo ime. Ob kliku na gumb, se pokaže meni, kjer se uporabnik lahko izpiše, ali pa pogleda svoj profil.
Pod navigacijsko vrstico je vrtiljak kratkih novic, ki vsakih pet sekund zamenja novico. S klikom na puščici levo oziroma desno lahko uporabnik menja novice. Pod vrtiljakom si lahko prebere še več novic, ki pa so bolj obsežne. Pod novicami si lahko pogleda predel namenjen sponzorjem, ki bi v teoriji podprli razvoj aplikacije. V nogi pa se nahaja več povezav do strani, kot so pomoč in kontakt. Na desni strani noge pa se uporabnika prosi, da aplikacijo deli na socialnih omrežjih, kot so Twitter, Facebook in Instagram, saj če bi hoteli, da bi aplikacija zaživela, bi jo bilo potrebno oglaševati na ostalih spletnih portalih.

5.1.3 dipl/public/shootingRangePage

Stran je namenjena temu, da si lahko uporabnik na hiter in enostaven način pregleda vsa slovenska strelišča in se, glede na zemljevid, odloči, kje bi bilo

najbolje trenirati disciplino, ki si jo želi. Ker pa je disciplin veliko in se strelišča med disciplinami zelo razlikujejo, saj se z nekaterimi orožji strelja na razdaljo manjšo od 20 metrov, z drugimi pa na razdaljo večjo od 300 metrov, smo ločili strelišča glede na nekaj kriterijev. Ker smo ljudje zelo vizualna bitja, smo vsak kriterij označili s svojo barvo. Zemljevid z označenimi vnešenimi strelišči je prikazan na sliki 5.3.

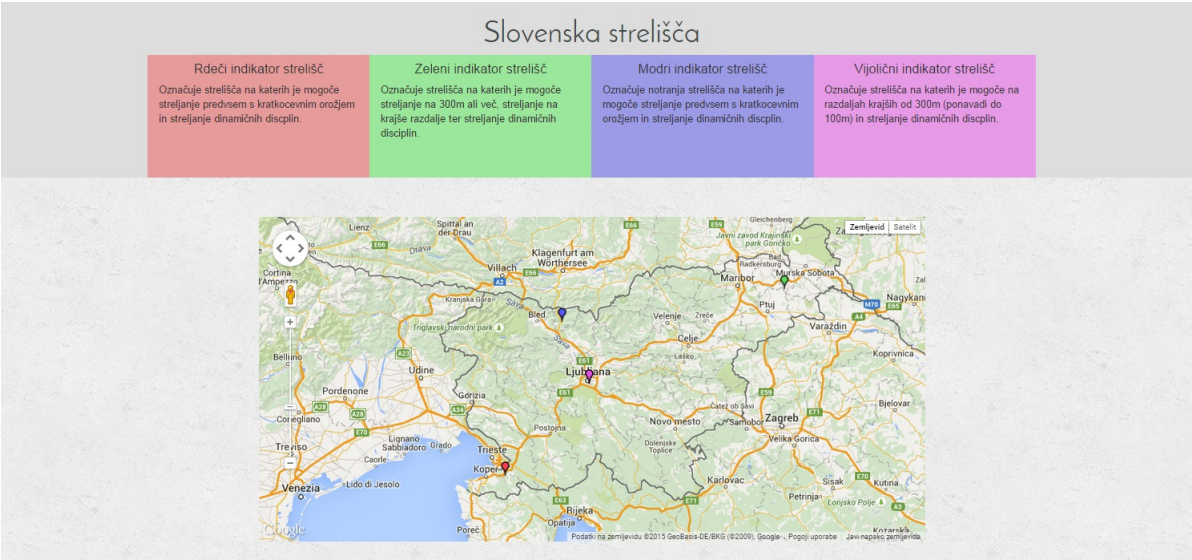
Rdeča barva označuje strelišča, na katerih je mogoče streljanje predvsem s kratkocevnim orožjem.

Zelena barva označuje strelišča, kjer je mogoče streljanje na več kot 300 metrov ter streljanje na krajše razdalje.

Modra barva označuje notranja strelišča, kjer je omogočeno streljanje s kratkocevnim orožjem.

Vijolična barva pa označuje strelišča, kjer je mogoče streljanje do 300m.

Zemljevid je vgrajen v spletno aplikacijo s pomočjo Google maps API. Omogoča tudi klik na posamezno strelišče. Ob kliku se pokaže pojavno okno, kjer so zapisane pomembne informacije o strelišču, kot je na primer naslov. Trenutno je na zemljevidu prikazana le peščica strelišč, saj podatkovne baze še nismo napolnili z vsemi strelišči in njihovimi podatki.



Slika 5.3: Prikaz slovenskih strelišč.

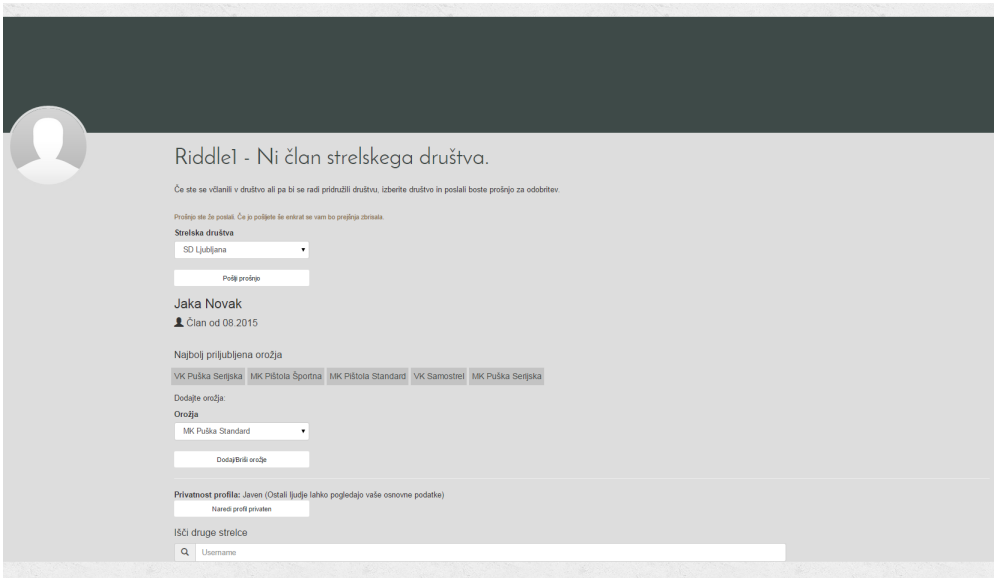
5.1.4 dipl/public/normPage

Preprosta stran, kjer uporabnik lahko pregleda, kakšne so v letošnjem letu norme za nastop na državnem prvenstvu republike Slovenije. Norme so razdeljene glede na orožje, za vsako orožje pa potem še glede na kategorije v katerih lahko strelci tekmujejo. Za vsako orožje je napisano, koliko strelov je potrebno ustreliti in koliko točk, posamezno ali ekipno, morajo strelci doseči, da se uvrstijo na državno prvenstvo. Posebnost te strani je gumb za animacijo poti do vrha. Ker je norm zelo veliko, je stran precej visoka in če se uporabniku ne da ročno pomikati do vrha strani, lahko klikne na gumb in se mu stran sama animirano premakne na vrh.

5.1.5 dipl/public/userPage

Do uporabniškega profila dostopamo prek gumba na desni zgornji strani glave. Primer uporabniškega profila je prikazan na sliki 5.4. Uporabniški profil je narejen zato, da ostali lahko vidijo določene informacije o uporabniku, sam pa jih lahko sebi ureja. Na vrhu se prikaže profilna slika, na desni pa se izpiše še uporabniško ime uporabnika. Če je uporabnik član strelskega društva, potem se mu poleg izpiše njegovo društvo, če pa ni član, pa se mu pojavi obrazec za pridružitev strelskemu društvu. Uporabnik si v seznamu izbere strelsko društvo, kateremu bi se želel pridružiti in s klikom na gumb pošlje prošnjo za pridružitev. Strelsko društvo mu potem bodisi prošnjo sprejme in ga s tem potrdi kot uradnega člana društva, ali pa ga zavrne. Pod izbiro strelskega orožja se izpiše še pravo ime in priimek uporabnika, pod tem pa njegova najbolj priljubljena orožja. Uporabnik lahko iz seznama izbere svoja najbolj priljubljena orožja in jih doda na seznam svojih najbolj priljubljenih. Orožja lahko iz priljubljenih tudi odstranimo s klikom na gumb *Dodaj/Briši orožje*. Pod tem se nahaja nastavitev privatnosti profila. Uporabnik ima lahko profil privaten, v tem primeru se njegovega profila ne bo videlo, ko ga bodo hoteli drugi uporabniki iskati. Če iščemo uporabnika, ki ima privaten profil, nam bo spletna aplikacija sporočila, da uporabnik ne

obstaja. Ravno to pa je naslednja funkcija, ki jo stran profila ponuja, iskanje drugih uporabnikov. Z vpisom uporabniškega imena v iskalnik lahko poiščem druge strelce in si ogledam njihove profile. Večina podatkov je za zunanje oglede nevidnih, prikaže se le uporabniško ime, ime strelskega društva, slika in pa najbolj priljubljena orožja.



Slika 5.4: Prikaz uporabniškega profila.

5.1.6 dipl/public/rulesPage

Na tej strani si uporabnik lahko prebere in prenese prevedena pravila mednarodne strelske zveze. Ob kliku na ikono, ki označuje PDF format dokumenta, se ta zasveti in začne se prenos dokumenta. Pravila so razdeljena v pravila za posamezna orožja ter splošna pravila.

5.1.7 dipl/public/targetPage

Če si uporabnik zaželi trenirati, vendar nima nobenih tarč, je prav ta stran to, kar potrebuje. Na tej strani ima uporabnik možnost izbrati in prenesti tarče za različna orožja, ki so prilagojene za tikanje na A4 papir. Stran s

tarčami, prilagojenimi za A4 papir, je prikazana na sliki 5.5. Te tarče sicer niso uradne in jih na tekmah ne morejo uporabiti, vendar pa so za trening ravno pravšnje. Tarče so razdeljene na tarče, ki se uporabljajo pri streljanju na 25 - 50 metrih in tiste, ki se uporabljajo pri streljanju na 50 -100 metrih.

Slika 5.5: Prikaz strani, kjer lahko uporabnik dobi strelske tarče.

5.1.8 dipl/public/dashBoard

Nadzorna plošča je najpomembnejša stran v tej spletni aplikaciji. Na njej si lahko uporabnik beleži svoje rezultate, pridruži razpisanim turnirjem in opravlja mnoge druge funkcionalnosti. Vsa interakcija na nadzorni plošči je narejena s tehnologijo AJAX, zato je nadzorna plošča tudi zelo hitra,, saj je potreba po izrisu HTML kode zelo majhna. Ker je prenosa HTML kode zelo malo ob menjavanju funkcionalnosti, je to dobro tudi za uporabnike, ki dostopajo prek telefonov in morajo podatke prenašati preko mobilnega omrežja. Stran sestavlja več zavihkov:

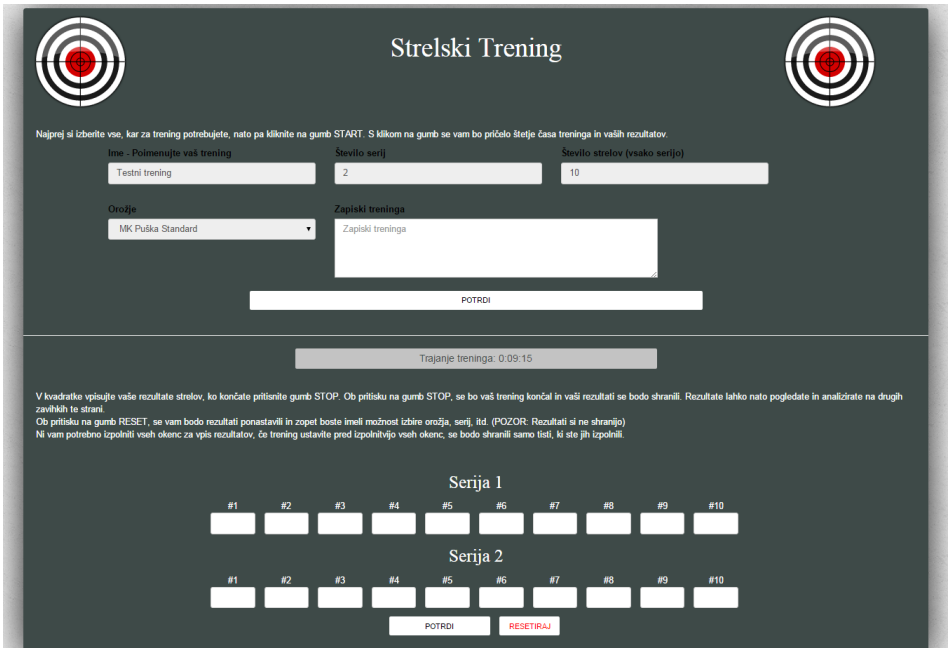
- **Opis**
  Opis je prvi zavihek, ki je odprt, ko stran prvič naložimo. Na glavni plošči, ki se nahaja desno od navigacijskega stolpca, se uporabniku prikaže kratek opis, kaj vse lahko na nadzorni plošči počne. Glavna

stvar, ki jo opis izpostavi, je grafični prikaz rezultatov, ki ga lahko pogledamo v zavihku pregled rezultatov.

- **Začni trening**

  Kot že samo ime zavihka pove, lahko tukaj uporabnik začne svoj strelski trening. Preden uporabnik začne s treningom, mora izpolniti obrazec za začetek treninga. Tudi vnosna polja tega obrazca se pregledajo z regularnimi izrazi, preden uporabnik začne s treningom. Vnosno polje ime sicer ni potrebno polje za začetek treninga, saj je trening lahko tudi brez imena, vendar pa uporabnika za to početje aplikacija opozori, saj neimenovanega treninga kasneje v zavihku *pregled treningov* ne moremo najti. Vpisati mora tudi število serij in število strelov. Obe polji se preverita z regularnim izrazom, če vsebujeta samo številke. Prav tako pa sta obe navzgor in navzdol omejeni. Čeprav načeloma omejitev pri treningu ne poznamo, sta številki omejeni, ker bi bilo vpisovanje veliko podatkov hkrati zelo nerodna zadeva. Število serij je omejeno med 1 in 10, število strelov na serijo pa med 1 in 15, kar je skupaj lahko kar 150 strelov. Uporabnik mora nato izbrati iz seznama še pravilno orožje in že je pripravljen na začetek treninga. Podano je tudi polje za zapiske, če bi si uporabnik želel zapisati kako beležko glede treninga.
  Ob kliku na gumb *Potrdi* se spodaj odpre nov razširjen predel, kjer se pokažejo okenca za vpis rezultatov strelov. Začne se šteti tudi čas trajanja treninga. Pod okenci se pojavita tudi dva nova gumba. Gumb *Resetiraj*, ki ponastavi vpisovanje strelov in podatke, ki so bili v njih vpisani. Prav tako ponastavi tudi čas treninga. Spletni brskalnik uporabnika ob pritisku gumba še enkrat vpraša, če si to zares želi narediti. S pritiskom na spodnji gumb *Potrdi* pa se konča uporabnikov trening in se shrani v podatkovno bazo. Ena izmed prednosti je ta, da za konec treninga ni potrebno izpolniti vseh okenc. Če je kako okno prazno, se ga ne šteje k skupnemu rezultatu. Po kliku in uspešni shranitvi v bazo, se na vrhu izpiše sporočilo, da je bil trening uspešno shranjen. Analize treningov si uporabnik potem lahko pogleda na drugih zavih-

kih. Ker je stran odzivno narejena, je primerna za uporabo z mobilnim telefonom, tako, da si strelec lahko na trening prinese mobilni telefon in sproti zapisuje rezultate. Slika 5.6 prikazuje nadzorno ploščo, kjer lahko uporabnik začne strelski trening.

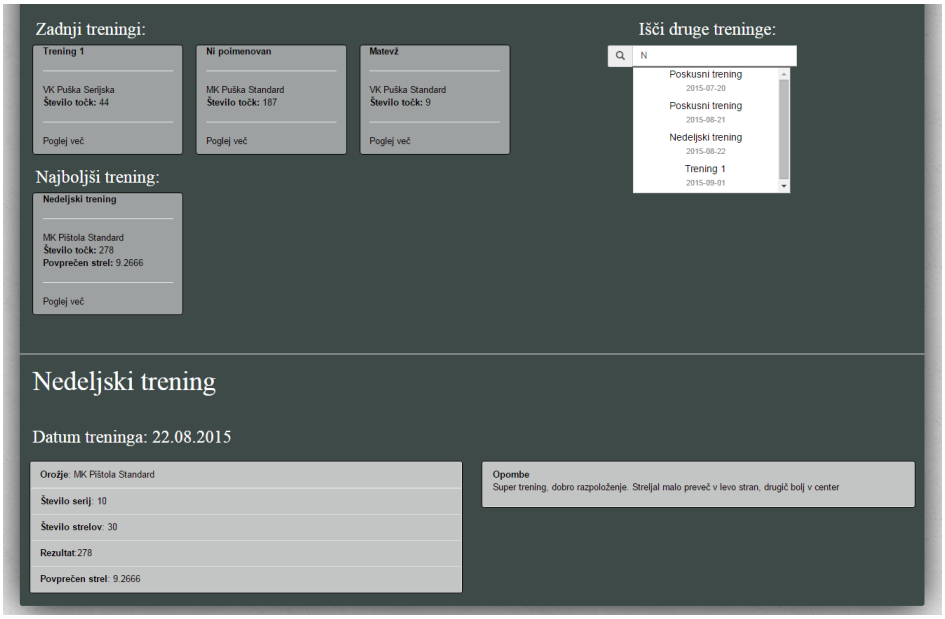


Slika 5.6: Prikaz dela nadzorne plošče, kjer lahko uporabnik začne strelski trening.

- **Pregled treningov**

  Ob odprtju zavihka se uporabniku že prikažejo ponujeni treningi. Slika 5.7 prikazuje del kontrolne plošče, ki omogoča pregled strelskih treningov uporabnika. V zgornji vrstici so zadnji treningi, ki jih je strelec izvedel. Za vsak trening se izpiše, kako je treningu ime, katero orožje je uporabljal, koliko točk je dosegel ter povezava, kjer si lahko pogleda več o treningu. Pod vrstico z zadnjimi treningi se nahaja vrstica z najboljšim treningom. Najboljši trening se izračuna glede na povprečje točk strelov, ki jih je uporabnik med treningom dosegel. Na desni strani pa se nahaja še iskanje treningov. Treninge lahko uporabnik išče glede

na njihova imena. Če treninga v bazi ne najde, se to izpiše, če pa se v bazi nahaja kak trening s takim imenom, pa ponudi seznam vseh treningov s takim imenom. Ob kliku na trening v iskalnem seznamu ali pa na povezavo *"poglej več treningov"*, ki so že prikazani, se pod tem razpre nov predel, kjer je za ta trening napisanih več informacij, vključno z opombami in datumom treninga.



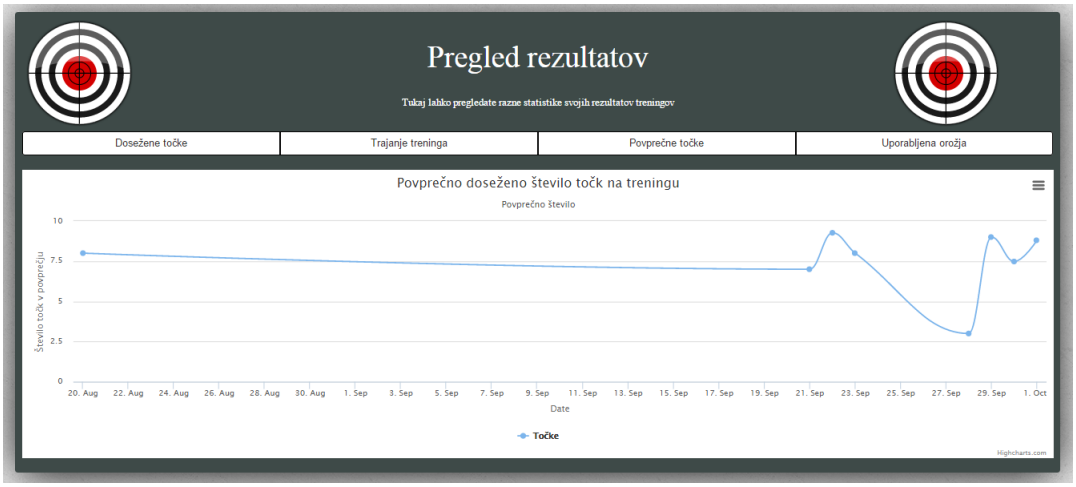
Slika 5.7: Prikaz dela nadzorne plošče, kjer lahko uporabnik pregleda svoje strelske treninge.

- **Pregled rezultatov**
  To je zavihek, ki je med vsemi najbolj grafično izpostavljen. Na tem zavihku si lahko uporabnik ogleda statistike svojih rezultatov treningov. Grafe si lahko uporabnik tudi prenese v različnih formatih, kot so PDF, JPEG, PNG. Ob odprtju zavihka se mu izpišejo štiri kategorije, ki si jih lahko pogleda, to so:

  - Dosežene točke
    Izriše se krivulja doseženih točk na treningu glede na datum. Gre

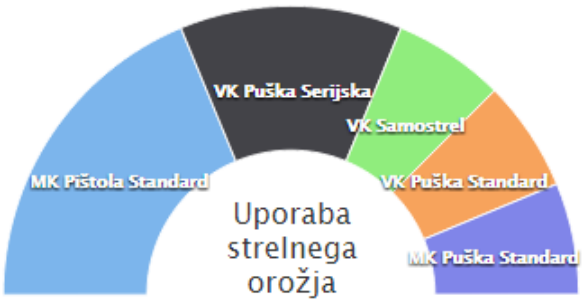
za absolutno število točk na treningu, tako, da ta graf morda ni najboljši za primerjavo uspešnosti treningov med sabo, saj ne gleda povprečnega doseženega števila točk. Pokaže pa vzdržljivost strelca in njegovo fizično kondicijo.

- Trajanje treninga
  Meri trajanje treninga v sekundah in glede na datum izriše krivuljo.
- Povprečne točke
  Verjetno najboljša metrika za merjenje uspešnosti treninga, saj je povprečje izračunano iz števila strelov in števila doseženih točk. Z dovolj podatki bi lahko glede na to napovedali nek trend gibanja grafa v prihodnosti. Slika 5.8 prikazuje graf doseženih povprečnih točk glede na datum.



Slika 5.8: Prikaz povprečnih točk doseženih na treningu glede na datum.

- Uporabljena orožja
  Slika 5.9 predstavlja prikaz povprečne rabe orožja na treningih.

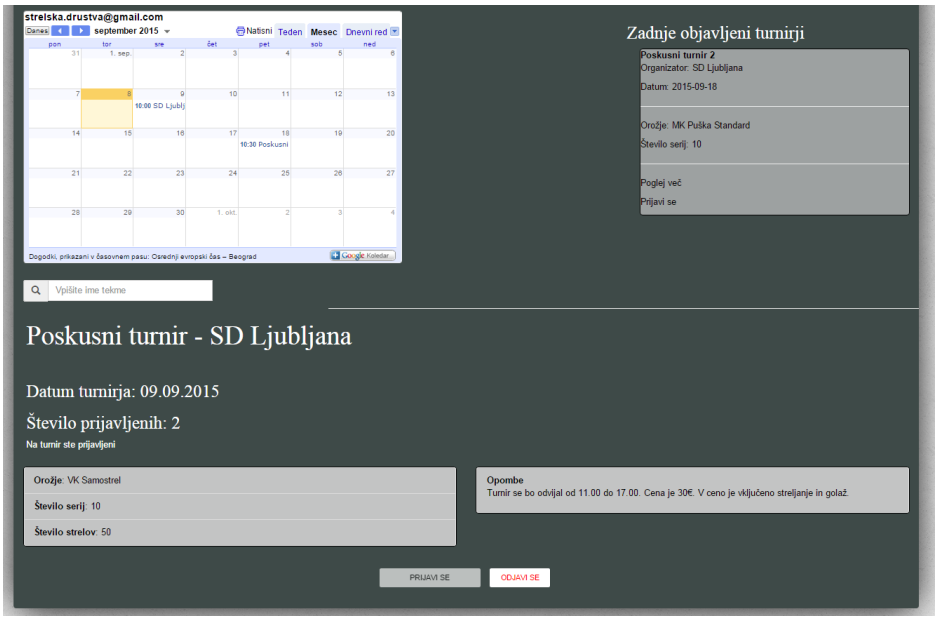


Slika 5.9: Prikaz povprečne rabe orožij na treningih.

- **Pregled tekmovanj**

  Podobno kot na zavihku za pregled treningov, je tudi tukaj na desni zgornji strani objavljen zadnji organiziran turnir. Za objavljen turnir piše, kdo je organizator, kdaj se bo izvajal in s katerim orožjem ter koliko bo serij. Pod tem se nahajata še dva gumba, kjer se uporabnik lahko prijavi na turnir ali pa pogleda več informacij o turnirju. Če je uporabnik že prijavljen, je gumb za prijavo zatemnjen. Na levi strani se nahaja koledar. Preko Google Calendar API se vsem uporabnikom spletne aplikacije prikaže koledar uradnega elektronskega poštnega naslova spletne aplikacije (strelska.drustva@gmail.com). Na koledarju so objavljeni vsi turnirji, ki so jih prijavila strelska društva. Kot uporabnik lahko ta koledar vidiš, lahko pa tudi klikneš na nek dogodek in prikazalo se bo pojavno okno. V pojavnem oknu se izpiše ime turnirja ter kdaj se dogaja. Uporabnik lahko nato pod koledarjem išče po imenih objavljenih turnirjev. Na sliki 5.10 je prikazan koledar spletne aplikacije.

  Ko uporabnik najde turnir, lahko nanj klikne, spodaj pa se mu odpre nov predel, ki se naloži s tehnologijo AJAX. V novem predelu se izpiše ime turnirja, organizator turnirja, datum turnirja ter število prijavljenih na turnir. Uprabnika tudi opozori, ali je na turnir že prijavljen.



Slika 5.10: Prikaz koledarja spletne aplikacije.

Če še ni, potem samo klikne gumb *PRIJAVI SE* in uporabnik bo prijavljen, če pa je že prijavljen, se lahko od turnirja odjavi z gumbom *ODJAVI SE*.

- **Android podpora**
  Ker je Android aplikacija omejena le na slikanje tarč in prek nje ne moremo vnesti rezultatov treningov, smo ravno za ta namen naredili spletno podporo Android aplikaciji. V tem zavihku povežemo Android aplikacijo in spletno aplikacijo. S pomočjo Android aplikacije slike shranimo na strežnik, v tem zavihku pa uporabniku slike pokažemo. Na vrhu plošče z vsebino je prikazan kratek opis, kaj na tem zavihku lahko naredimo. Pod tem opisom pa se nahajajo slike, ki jih je uporabnik naložil na strežnik. Slike so razporejene po vrsti, glede na to, kdaj so bile naložene na strežnik. Nad vsako sliko se prikaže njeno ime, pod sliko pa se nahaja gumb *Zapiši točke*. Ob kliku na gumb se s pomočjo tehnologije AJAX v spodnji predel naloži obrazec za zapisovanje rezultatov. Obrazec je podoben tistemu, s katerim začnemo in shranimo

trening. Tudi tukaj mora uporabnik vpisati ime treninga, število serij ter število strelov, ki jih je imel. Orožje lahko izbere iz seznama. Ob kliku na gumb se mu pokažejo polja, kamor vpisuje rezultate. Čisto spodaj, pod polji za vnos rezultatov, se nahaja še slika tarče, ki jo je strelec izbral. Zgoraj, kjer strelec izbira tarče, so slike tarč majhne, tukaj pa je izbrana tarča prikazana v večjem formatu, da lahko strelec brez težav vidi, kje v tarči se nahajajo zadetki. Na sliki 5.11 je prikazan seznam slik, ki jih je strelec naložil na strežnik.

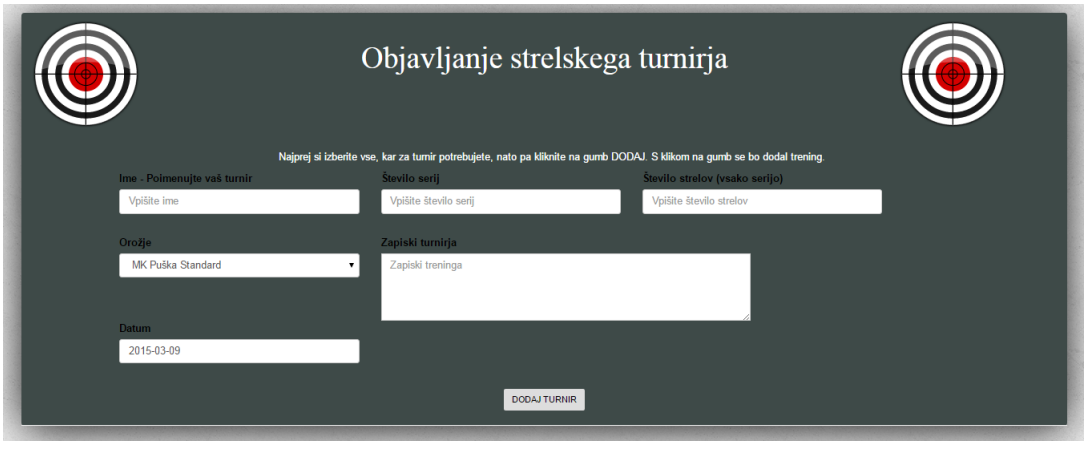
Slika 5.11: Prikaz strelčevih slik v zavihku "Android podpora".

5.2 Funkcionalnosti strelskih društev

Funkcionalnosti strelskih društev so po večini enake, kot pri uporabniku, zato bomo v tem poglavju opisali samo tiste, ki se razlikujejo. Strelska društva imajo drugačno nadzorno ploščo kot uporabniki, kar jim omogoča akcije, ki jih uporabnik ne more narediti, kot je na primer objava turnirja.

5.2.1 Objavi turnir

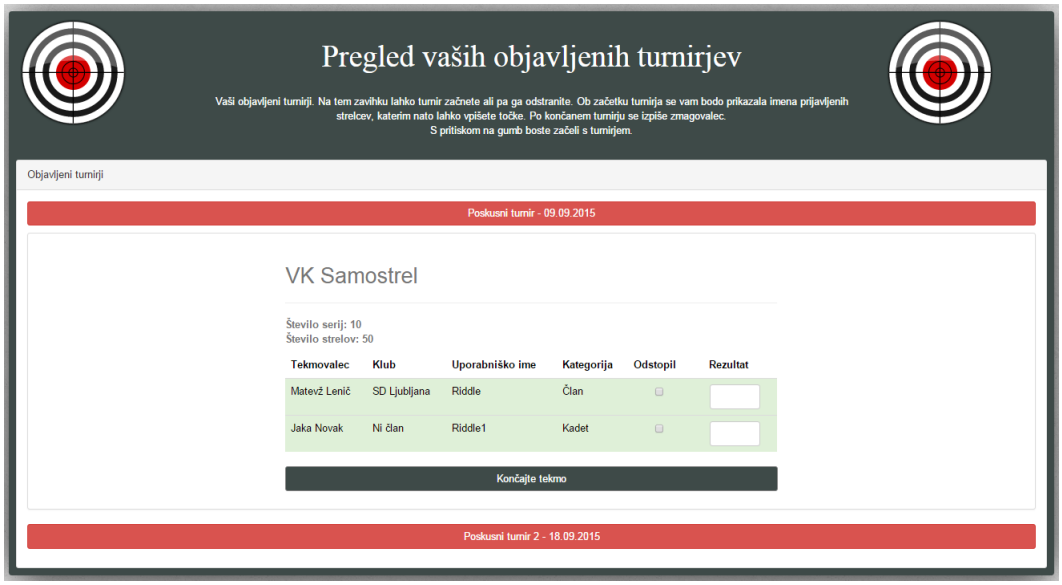
Na tem zavihku lahko strelsko društvo objavi uradni turnir za vse člane spletne aplikacije. Slika 5.12 prikazuje obrazec za objavo novega turnirja. Grafični vmesnik je zelo podoben tistemu za dodajanje turnirja pri uporabniku. Turnirju je potrebno dati ime, določiti število serij in strelov ter orožje in datum. Dodajanje zapiskov je opcijsko, vendar je za turnir zaželjeno, da se v zapiske napišejo stvari, kot so prijavnina ter kraj izvajanja. Za pravilnost vnešenih podatkov skrbi JavaScript s pomočjo regularnih izrazov. Število serij je med 1 in 10, število strelov pa med 1 in 15. Izbira datuma je narejena s pomočjo 'bootstrap-datepicker' knjižnice. Ob kliku na gumb *DODAJ TURNIR* se v podatkovno bazo zapiše turnir in uporabniki se že lahko začnejo prijavljati.



Slika 5.12: Obrazec za objavo novega turnirja.

5.2.2 Začni turnir

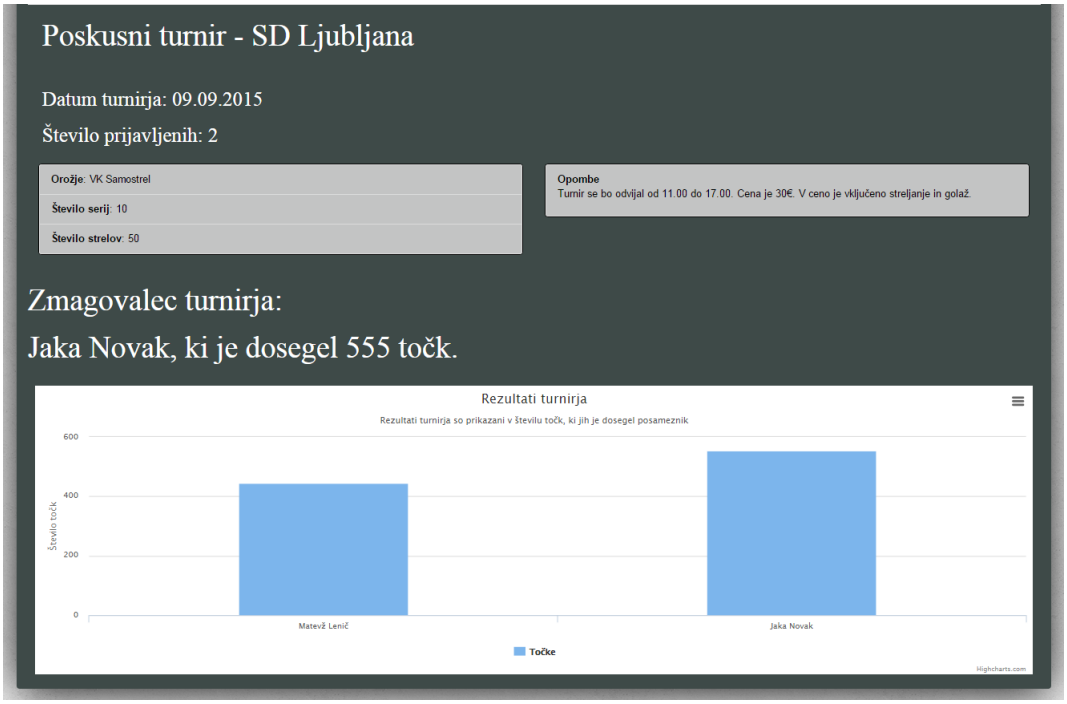
Ob kliku na ta zavihek se s pomočjo AJAXa prenesejo v pogled vsi turnirji, ki jih je neko strelsko društvo razpisalo. V seznamu se pojavijo imena vseh turnirjev in datumi njihove izvedbe. Ob kliku na gumb, ki označuje turnir, se bo turnir pričel. Na vrhu se najprej, kot je razvidno iz slike 5.13, izpiše orožje, s katerim tekmovalci tekmujejo, nato število serij in strelov. Pod tem pa se začne seznam tekmovalcev, ki na tekmi sodelujejo. Za vsakega tekmovalca se izpiše njegovo ime, klub za katerega strelja, uporabniško ime in kategorija. Ker se v nekem resničnem scenariju lahko zgodi, da se nek prijavljeni uporabnik na turnirju ne pojavi, ga moramo biti sposobni odstraniti iz tekmovanja. S klikom na polje *Odstopil*, se uporabnik, ki ni prišel, blokira, tako, da se mu ne more več vpisati rezultata, razen če ponovno kliknemo na to polje. Ko se turnir konča, je potrebno v aplikacijo zapisati rezultate vseh strelcev in nato klikniti gumb *Končajte tekmo*. S tem se bo turnir shranil v bazo in se odstranil s seznama naših turnirjev. Rezultate turnirja si potem lahko pogleda v zavihku *"Preglej rezultate turnirja"*.



Slika 5.13: Prikaz pogleda, ki pokaže pregled naših turnirjev.

5.2.3 Poglej rezultate turnirja

Ob kliku na ta zavihek se prikaže zelo preprost vmesnik, ki vsebuje samo kratek opis in eno polje za iskanje. V iskalnem polju poiščemo enega izmed že končanih turnirjev in kliknemo nanj. Pod iskalnim poljem se razširi novo polje, ki vsebuje podrobnosti iskanega turnirja. Izpišejo se klasične stvari, kot so ime, orožje in število serij. Izpiše pa se tudi zmagovalec turnirja, skupaj s stolpčnim diagramom, ki prikazuje rezultate turnirja glede na število doseženih točk. Tisti z največ doseženimi točkami je zmagovalec. Slika 5.14 prikazuje pregled rezultatov turnirja.



Slika 5.14: Pogled rezultatov turnirja skupaj s podrobnostmi in zmagovalcem.

5.2.4 Člani strelskega društva

To je zavihek s preprosto vsebino, ki ga mora imeti vsako društvo. S pomočjo tega zavihka lahko strelsko društvo na spletni aplikaciji vodi evidenco svojih

članov. V preprosti tabeli se izpišejo imena, kategorija in datum pridružitve za vse člane. Na sliki 5.15 se nahaja prikaz članov strelskega društva.

Slika 5.15: Prikaz vseh članov strelskega društva.

5.2.5 Sprejmi člane

Kot smo omenili v poglavju 5.1.5, se lahko uporabnik, ki ni član nobenega društva, prijavi v društvo. To lahko naredi s klikom na gumb na svojem profilu. Po tem, ko uporabnik zaprosi za članstvo v strelskem društvu, se ta zahtevek shrani v podatkovno bazo in nato prikaže izbranemu strelskemu društvu v tem zavihku. Tudi v tem zavihku se prikaže preprosta tabela uporabnikov, ki se hočejo pridružiti strelskemu društvu. Primer je prikazan na sliki 5.16. V tabeli imamo zopet podatke o imenu, kategoriji ter pridružitvi. Imamo tudi dve dodatni polji, in sicer *Sprejmi* in *Zavrni*. Če se za uporabnika pritisne gumb *Sprejmi*, potem uporabnik uradno postane član tega strelskega društva in je takoj dodan na seznam članov. Če pa se klikne gumb *Zavrni*, potem se uporabniku sprosti prijava v to strelsko društvo in se ga iz tega seznama zbriše.

Slika 5.16: Prikaz vseh članov, ki se želijo pridružiti temu strelskemu društvu.

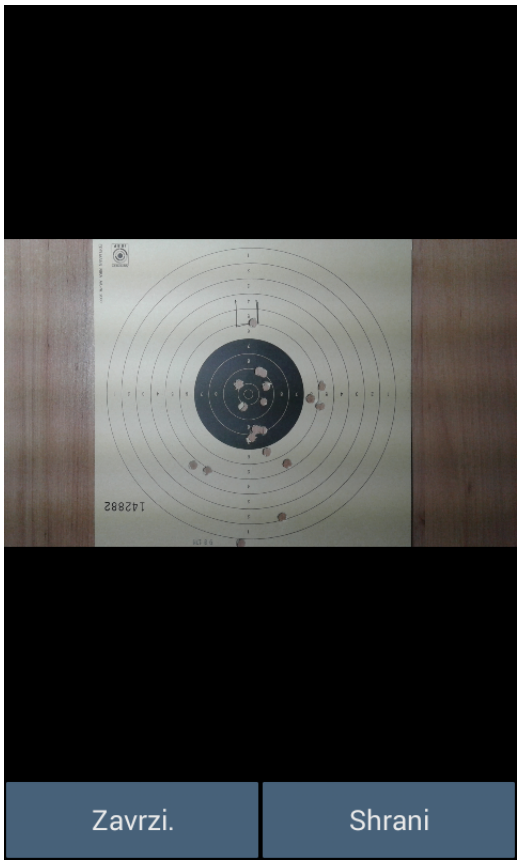
5.3 Funkcionalnosti Android aplikacije

Android aplikacija je namenjena samo uporabnikom spletne aplikacije *"Združena strelska društva"*, saj morajo biti uporabniki za njeno uporabo registrirani na spletni aplikaciji. Aplikacijo lahko uporabljajo samo strelci in ne strelska društva. Razlog za to je, da strelska društva na spletni aplikaciji ne morejo shranjevati treningov. Ker pa je Android aplikacija v celoti namenjena lažjemu shrajevanju treningov, jo lahko tako uporabljajo samo strelci. Aplikacijo lahko tudi glede funkcionalnosti razdelimo na dva dela:

- **LoginActivity**
  V tem pogledu se lahko uporabnik vpiše v aplikacijo. Uporabnik vpiše svoje uporabniško ime in geslo in pritisne na gumb *PRIJAVA*. Če sta podatka pravilna, se uporabnika preusmeri na naslednji pogled, če pa sta napačna, se pokaže pojavno okno, v katerem se uporabniku izpiše opozorilo. Na sliki 5.17 je prikazan pogled vpisa s pojavnim oknom, ki se pojavi, če uporabnik poda napačne podatke.

Slika 5.17: Prikaz sporočila v Android aplikaciji ob napačno podanih vpisnih podatkih.

- **HelloActivity**
  Tudi uporaba te aktivnosti je precej preprosta. Uporabnik lahko pritisne na gumb *SLIKAJ*. Ob pritisku na ta gumb se odpre kamera, kjer lahko uporabnik zajame sliko, ki jo želi poslati na strežnik. Po zajemu slike, se slika prikaže na zaslonu, pod njo pa ima uporabnik dva gumba, kot je prikazano na sliki 5.18. Ob kliku na gumb *Zavrzi*, se slika ne shrani na strežnik in uporabnik lahko naredi novo. Ob kliku na gumb *Shrani*, pa se uporabnika usmeri nazaj na aktivnost, slika pošlje na strežnik, kjer se tudi shrani. Če je bila slika na strežniku uspešno shranjena, se bo uporabniku na tej aktivnosti prikazalo pojavno okno, kjer piše, da je bila slika uspešno naložena.



Slika 5.18: Prikaz zajete slike in izbire, ki jo ima uporabnik.

Poglavje 6

Sklep in nadaljnje delo

V sklopu diplomskega dela smo si zadali cilj, da izdelamo platformo, ki bi pomagala strelstvu kot športni panogi. Osredotočili smo se na strelce in strelska društva. Ta cilj smo dosegli tako, da smo strelcem in strelskim društvom ponudili neko skupno platformo, ki je prva taka na tržišču.
V prvi fazi diplomskega dela smo razvili spletno aplikacijo. Za razvoj smo uporabili jezik PHP in ogrodje Laravel. Spletna aplikacija uporabniku ponuja več orodij, ki mu olajšajo treninge in tekme. Strelec lahko na svoji nadzorni plošči prične trening in si beleži rezultate treninga. Aplikacija mu omogoča tudi pregled rezultatov svojih treningov ter primerjavo različnih metrik skozi čas. Prek aplikacije se lahko strelec prijavi na turnir, ki ga organizira strelsko društvo. Tudi strelska društva imajo nekaj unikatnih orodij, ki jim pomagajo upravljati s turnirji in njihovimi člani. Strelsko društvo lahko razpiše turnir in pregleduje rezultate že končanih turnirjev. Poleg tega pa lahko prek spletne aplikacije sprejema tudi nove člane.
V drugi fazi diplomskega dela pa smo razvili mobilno aplikacijo, ki strelcem pomaga pri njihovem treningu. Mobilna aplikacija omogoča slikanje tarč in njihovo pošiljanje ter shranjevanje na strežnik. Strelec si lahko kasneje to sliko v spletni aplikaciji ogleda in si zapiše rezultate treninga.
Razvili smo torej aplikacijo, ki vsebuje vse, kar smo si pred začetkom razvoja zamislili. Menimo, da bo v celoti zgrajena aplikacija pomagala strelcem pri

njihovih treningih in strelskim društvom pri njihovi organizaciji turnirjev ter, da bo aplikacija pripomogla k še večji prepoznavnosti strelstva kot športa v Sloveniji.

Ob sami izdelavi spletne in mobilne aplikacije smo si zamislili še več izboljšav, ki bi jih lahko implementirali. Ker se danes na spletne strani dostopa z veliko različnimi napravami, je potrebno zagotoviti dobro uporabniško izkušnjo pri uporabi na vseh napravah oziroma odzivnost spletne aplikacije. Prvo stran ter nadzorno ploščo smo naredili popolnoma odzivno, saj sta to strani, do katerih bo največ uporabnikov dostopalo. Ostale strani pa niso popolnoma odzivne in se ob spremembi višine ali širine okna, njihov izgled lahko pokvari. Tako je ena izmed možnih izboljšav dodajanje oziroma izboljšanje odzivnosti ostalih strani. Ker je mobilna aplikacija precej preprosta, imamo na voljo veliko izboljšav oziroma razširitev aplikacije. Na mobilno aplikacijo bi lahko dodali še ostale funkcionalnosti spletne aplikacije, kot sta nadzorna plošča ter stran z novicami. Poleg tega bi uporabnost mobilne aplikacije povečali tudi z avtomatskim zaznavanjem strelskih lukenj. Aplikacija bi lahko ob zajemu slike tarče izračunala, kje se nahajajo luknje in to avtomatsko prenesla v podatkovno bazo na strežniku, skupaj s sliko. To bi nam prineslo še boljšo uporabniško izkušnjo.

Kljub vsem tem izboljšavam pa menimo, da sta spletna in mobilna aplikacija zadostili ciljem, ki smo si jih zadali pred začetkom razvoja.

Literatura

[1] Programska knjižnica. Dosegljivo: http://www.webopedia.com/TERM/L/library.html.

[2] Programsko ogrodje. Dosegljivo: http://whatis.techtarget.com/definition/framework.

[3] Laravel dokumentacija. Dosegljivo: http://laravel.com/docs/5.1

[4] Označevalni jezik za oblikovanje večpredstavnostnih dokumentov. Dosegljivo: http://www.yourhtmlsource.com/starthere/whatishtml.html

[5] World Wide Web Consortium. Dosegljivo: http://www.w3.org/.

[6] Kaskadne slogovne pole (*angl. Cascading style sheets*). Dosegljivo: http://www.w3.org/Style/CSS/Overview.en.html

[7] The PHP Group. Dosegljivo: http://php.net/copyright.php

[8] PHP. Dosegljivo: https://en.wikipedia.org/wiki/PHP.

[9] Število strani z strežniško tehnologijo PHP. Dosegljivo: http://w3techs.com/technologies/overview/programming_language/all

[10] JavaScript. Dosegljivo: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction

[11] Sun Microsystems. Dosegljivo:
https://en.wikipedia.org/wiki/Sun_Microsystems

[12] Programski jezik Java. Dosegljivo:
http://searchsoa.techtarget.com/definition/Java

[13] Razširjenost jezika Java. Dosegljivo:
http://w3techs.com/diagram/market_technology/pl-java

[14] MySQL .Dosegljivo:
http://dev.mysql.com/doc/

[15] Ogrodje Bootstrap. Dosegljivo:
http://getbootstrap.com/

[16] Knjižnica Jquery. Dosegljivo:
https://en.wikipedia.org/wiki/JQuery

[17] Podjetja, ki uporabljajo knjižnico Highcharts. Dosegljvo:
http://shop.highsoft.com/highcharts.html

[18] Laravel ogrodje. Dosegljivo:
http://laravel.com/

[19] Raziskava o PHP ogrodjih. Dosegljivo: http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/

[20] ORM. Dosegljivo:
https://www.techopedia.com/definition/24200/object-relational-mapping–orm

[21] MVC. Dosegljivo:
http://laravelbook.com/laravel-architecture/

[22] Orodje XAMPP. Dosegljivo:
https://www.apachefriends.org/about.html

[23] Android Studio. Dosegljivo:
http://developer.android.com/tools/studio/index.html

[24] Orodje HeidiSQL. Dosegljivo:
http://www.heidisql.com/#featurelist

[25] doc. Dr. Tomaž Hovelja. Uporabniška vrzel. Dosegljivo:
https://ucilnica.fri.uni-https://ucilnica.fri.uni-lj.si/pluginfile.php/53993/mod_folder/content/0/Organizacija%20in%20management%202015%20vse%20prosojnice.pdf

[26] T. Felke-Morris: Basics of web design: HTML5 & CSS3, Addison-Wesley, 2012

[27] doc. Dr. Aleš Smrdel. Oblikovanje spletne strani. Dosegljivo:
https://ucilnica.fri.uni-lj.si/pluginfile.php/39507/mod_label/intro/P02-OrodjaOblikovanjeHTML-HO-1.pdf

[28] Predstavitev Laravel arhitekture. Dosegljivo:
http://community.codeup.com/

[29] Konceptualni, logični in fizični podatkovni model. Dosegljivo:
http://www.1keydata.com/datawarehousing/data-modeling-levels.html

[30] Odstotek mobilnih naprav, ki uporabljajo Android. Dosegljivo:
http://gs.statcounter.com/#mobile_os-ww-daily-20150725-20150904

[31] Apache HTTP komponente. Dosegljivo:
http://hc.apache.org/httpcomponents-client-ga/tutorial/html/fundamentals.html#d5e37